

305-CD-007-001

EOSDIS Core System Project

Release A SDPS Data Management Subsystem Design Specification for the ECS Project

July 1995

Hughes Information Technology Corporation
Landover, MD

Release A SDPS Data Management Subsystem Design Specification for the ECS Project

July 1995

Prepared Under Contract NAS5-60000
CDRL Item #046

SUBMITTED BY

<u>Parag N. Ambardekar /s/</u>	<u>7/27/95</u>
Parag Ambardekar, Release A CCB Chairman	Date
EOSDIS Core System Project	

Hughes Information Technology Corporation
Landover, Maryland

This page intentionally left blank.

Preface

This document is one of sixteen comprising the detailed design specifications of the SDPS and CSMS subsystem for Release A of the ECS project. A complete list of the design specification documents is given below. Of particular interest are documents number 305-CD-004, which provides an overview of the subsystems and 305-CD-018, the Data Dictionary, for those reviewing the object models in detail. A Release A SDPS and CSMS CDR Review Guide (510-TP-002) is also available.

The SDPS and CSMS subsystem design specification documents for Release A of the ECS Project include:

- 305-CD-004 Release A Overview of the SDPS and CSMS Segment System Design Specification
- 305-CD-005 Release A SDPS Client Subsystem Design Specification
- 305-CD-006 Release A SDPS Interoperability Subsystem Design Specification
- 305-CD-007 Release A SDPS Data Management Subsystem Design Specification
- 305-CD-008 Release A SDPS Data Server Subsystem Design Specification
- 305-CD-009 Release A SDPS Ingest Subsystem Design Specification
- 305-CD-010 Release A SDPS Planning Subsystem Design Specification
- 305-CD-011 Release A SDPS Data Processing Subsystem Design Specification
- 305-CD-012 Release A CSMS Segment Communications Subsystem Design Specification
- 305-CD-013 Release A CSMS Segment Systems Management Subsystem Design Specification
- 305-CD-014 Release A GSFC Distributed Active Archive Center Implementation
- 305-CD-015 Release A LaRC Distributed Active Archive Center Implementation
- 305-CD-016 Release A MSFC Distributed Active Archive Center Implementation
- 305-CD-017 Release A EROS Data Center Distributed Active Archive Center Implementation
- 305-CD-018 Release A Data Dictionary for Subsystem Design Specification
- 305-CD-019 Release A System Monitoring and Coordination Center Implementation

Object models presented in this document have been exported directly from CASE tools and in some cases contain too much detail to be easily readable within hard copy page constraints. The reader is encouraged to view these drawings on line using the Portable Document Format (PDF) electronic copy available via the ECS Data Handling System (ECS) at URL <http://edhs1.gsfc.nasa.gov>.

This document is a contract deliverable with an approval code 2. As such, it does not require formal Government approval, however, the Government reserves the right to request changes within 45 days of the initial submittal. Once approved, contractor changes to this document are handled in accordance with Class I and Class II change control requirements described in the EOS Configuration Management Plan, and changes to this document shall be made by Document Change Notice (DCN) or by complete revision.

Any questions should be addressed to:

Data Management Office
The ECS Project Office
Hughes Information Technology Corporation
1616 McCormick Drive
Landover, MD 20785

This page intentionally left blank.

Abstract

This document presents the design of the Data Management Subsystem of the Earth Observing System Data and Information System (EOSDIS) Core System (ECS). It defines the Data Management Subsystem's Release A CSCI and HWCI structures, as well as subsystem design based on Level 4 requirements.

Keywords: SDPS, Data Management, CSCI, HWCI, V0 Gateway, Gateway, ODL, V0 Client

This page intentionally left blank.

Change Information Page

List of Effective Pages			
Page Number		Issue	
Title		Original	
iii through xii		Original	
1-1 through 1-2		Original	
2-1 through 2-4		Original	
3-1 through 3-4		Original	
4-1 through 4-50		Original	
5-1 through 5-8		Original	
AB-1 through AB-6		Original	
Document History			
Document Number	Status/Issue	Publication Date	CCR Number
305-CD-007-001	Original	July 1995	95-0553

This page intentionally left blank.

Contents

Preface

Abstract

1. Introduction

1.1	Identification	1-1
1.2	Scope	1-1
1.3	Document Organization	1-1
1.4	Status and Schedule	1-1

2. Related Documents

2.1	Parent Documents	2-1
2.2	Applicable Documents	2-1
2.3	Information Documents Not Referenced	2-2

3. Release A SDPS Data Management Subsystem Overview

3.1	Subsystem Overview	3-1
3.2	Subsystem Structure	3-1
3.3	Subsystem Design Rationale	3-1

4. GTWAY - Version 0 Gateway CSCI

4.1	CSCI Overview	4-1
4.2	CSCI Context	4-1
4.3	CSCI Object Model	4-1
4.3.1	GTWAY Request Processing Object Model	4-1
4.3.1.1	DmGwDirectory Request Class	4-3
4.3.1.2	DmGwInvRequests Class	4-5
4.3.1.3	DmGwProductRequest Class	4-9
4.3.1.4	DmGwRequestList Class	4-12
4.3.1.5	DmGwV0BrowseRequest Class	4-13
4.3.1.6	DmGwV0Request Class	4-16
4.3.1.7	V0ServerBackEnd Class	4-19
4.3.1.8	V0ServerFrontEnd Class	4-19
4.3.2	GTWAY Data Server Interface Object Model	4-19

4.3.2.1	DmGwAcquireRequest Class	4-21
4.3.2.2	DmGwBrowseRequest Class	4-22
4.3.2.3	DmGwDistribution Class.....	4-23
4.3.2.4	DmGwGateWayCollector Class	4-24
4.3.2.5	DmGwGateWayDescriptor Class	4-25
4.3.2.6	DmGwInvESDTReference Class.....	4-28
4.3.2.7	DmGwInvQuery Class.....	4-29
4.3.2.8	DmGwInvSearchRequest Class	4-30
4.3.2.9	DmGwMediaInfo Class	4-32
4.3.2.10	DsCIDescriptor Class.....	4-32
4.3.2.11	DsCIESDTReference Class	4-33
4.3.2.12	DsCIESDTReferenceCollector Class	4-33
4.3.2.13	DsCIQuery Class	4-33
4.3.2.14	DsCIRequest Class.....	4-34
4.3.3	GTWAY Persistent Data Object Model	4-34
4.3.3.1	DmGwBoundingCoordinates Class	4-36
4.3.3.2	DmGwDataCollection Class.....	4-37
4.3.3.3	DmGwDataCollectionMap Class.....	4-39
4.3.3.4	DmGwFieldCampaign Class	4-39
4.3.3.5	DmGwFieldCampaignMap Class	4-39
4.3.3.6	DmGwGeophysicalParameter Class.....	4-40
4.3.3.7	DmGwGeophysicalParameterMap Class.....	4-40
4.3.3.8	DmGWGranuleIdURMap Class	4-41
4.3.3.9	DmGwLocalityNameMap Class.....	4-41
4.3.3.10	DmGwMap Class.....	4-42
4.3.3.11	DmGwPlatformMap Class.....	4-42
4.3.3.12	DmGwSensorMap Class.....	4-43
4.3.3.13	DmGwSensorPlatform Class	4-43
4.3.3.14	DmGwStatusCodeMap Class.....	4-44
4.3.3.15	DmGwV0Requests Class.....	4-44
4.3.3.16	DmGwV0StatusMessage Class	4-45
4.3.4	GTWAY V0 ECS Mapping Service Object Model.....	4-45
4.3.4.1	DmGwV0ECSECSMapper Class	4-45
4.4	GTWAY - Version 0 Gateway CSCI Structure.....	4-50

5. DMGHW - Data Management HWCI

5.1	HW Design Drivers.....	5-1
5.1.1	Key Trade-off Studies and Prototypes.....	5-1
5.1.1.1	Prototype Studies	5-2

5.1.2	Sizing and Performance Analysis	5-2
5.1.2.1	HWCI Alternatives	5-3
5.1.3	Scalability, Evolvability and Migration to Release B.....	5-3
5.2	HWCI Structure	5-4
5.2.1	Connectivity	5-4
5.2.2	HWCI Components.....	5-5
5.3	Failover and Recovery Strategy.....	5-5
5.3.1	Server Hardware Failure Recovery.....	5-6
5.3.2	DBMS Failure Recovery.....	5-7
5.3.3	Network Recovery	5-8

Figures

3.2-1.	Data Management Subsystem Context	3-2
4.3.1-1.	DMGW-Requests Object Model Diagram.....	4-2
4.3.2-1.	DMGWDataServerIf Object Model Diagram.....	4-20
4.3.3-1.	DMGWPersistentData Object Model Diagram	4-35
4.3.4-1.	DMGWV0ECSEMapper Object Model Diagram.....	4-46
5.2-1.	Data Management HWCI	5-5
5.2.1-1.	Data Management Network Connectivity	5-6

Tables

3.2-1.	Subsystem Interfaces	3-3
4.4-1.	Gateway CSCI Components	4-50
5.2.2-1.	Data Management HWCI Components	5-6

Abbreviations and Acronyms

1. Introduction

1.1 Identification

This Release A SDPS Data Management Subsystem Design Specification for the ECS Project, Contract Data Requirement List (CDRL) Item 046, with requirements specified in Data Item Description (DID) 305/DV2, is a required deliverable under the Earth Observing System Data and Information System (EOSDIS) Core System (ECS), Contract NAS5-60000. This publication is part of a series of documents comprising the Science and Communications Development Office design specification for the Communications and System Management Segment (CSMS) and the Science and Data Processing Subsystem (SDPS) for Release A.

1.2 Scope

The Release A SDPS Data Management Subsystem Design Specification defines the progress of the design. It defines the Release A SDPS Data Management Subsystem computer software and hardware architectural design, as well as subsystem design based on Level 4 requirements.

This subsystem is on an incremental development track. It is released and reviewed in the form of Evaluation Packages (EP), and is therefore not part of the formal Release A Critical Design Review. The overview material for these components has been included in this document for information purposes only.

This document reflects the June 21, 1995 Technical Baseline maintained by the contractor configuration control board in accordance with the ECS Technical Direction No. 11 dated December 6, 1994.

1.3 Document Organization

The document is organized to describe the Data Management Subsystem design as follows:

Section 1 provides information regarding the identification, scope, status, and organization of this document.

Section 2 provides a listing of related documents which were used as source information for this document.

Section 3 provides an overview of the Subsystem, focusing on the high-level design concept. The section provides general background and context information for the subsystem.

Section 4 contains the structure of the computer software configuration items (CSCI) comprising the Data Management Subsystem.

Section 5 contains the hardware configuration item (HWCI) design of the Data Management Subsystem. This includes the hardware design for Interoperability Subsystem.

The section Abbreviations and Acronyms contains an alphabetized list of the definitions for abbreviations and acronyms used in this document.

1.4 Status and Schedule

This submittal of DID 305/DV3 meets the milestone specified in the Contract Data Requirements List (CDRL) of NASA Contract NAS5-60000. The submittal was reviewed during the SDPS Preliminary Design Review (PDR) and reflects changes to the design which resulted from that review. The PDR also triggered a number of follow up actions in response to Review Item Discrepancies (RID) the results of which are incorporated into the Critical Design Review (CDR) version of this document and subsystem design to be demonstrated at EP6.

2. Related Documents

2.1 Parent Documents

The parent document is the document from which the scope and content of this Data Management Subsystem Design Specification is derived.

194-207-SE1-001 System Design Specification for the ECS Project

2.2 Applicable Documents

The following documents are referenced within this SDPS Design Specification, or are directly applicable, or contain policies or other directive matters that are binding upon the content of this document.

60-TP-004-001	User Pull Analysis Notebook
209-CD-001-001	Interface Control Document Between EOSDIS Core System (ECS) and the NASA Science Internet
209-CD-002-001	Interface Control Document Between EOSDIS Core System (ECS) and ASTER Ground Data System
209-CD-003-001	Interface Control Document Between EOSDIS Core System (ECS) and EOS-AM Project for AM-1 Spacecraft Analysis Software
209-CD-004-001	Data Format Control Document for the Earth Observing System (EOS) AM-1 Project Data Base
209-CD-005-001	Interface Control Document Between EOSDIS Core System (ECS) and Science Computing Facilities (SCF)
209-CD-006-001	Interface Control Document Between EOSDIS Core System (ECS) and National Oceanic and Atmospheric Administration (NOAA) Affiliated Data Center (ADC)
209-CD-007-001	Interface Control Document Between EOSDIS Core System (ECS) and TRMM Science Data and Information System (TSDIS)
209-CD-008-001	Interface Control Document Between EOSDIS Core System (ECS) and the Goddard Space Flight Center (GSFC) Distributed Active Archive Center (DAAC)
209-CD-009-001	Interface Control Document Between EOSDIS Core System (ECS) and the Marshall Space Flight Center (MSFC) Distributed Active Archive Center (DAAC)
209-CD-011-001	Interface Control Document Between EOSDIS Core System (ECS) and the Version 0 System
305-CD-003-002	Communications and System Management Segment (CSMS) Design Specification for the ECS Project
308-CD-001-003	Software Development Plan for the ECS Project

313-CD-004-001	Release A CSMS/SDPS Internal Interface Control Document for the ECS Project
423-41-03	Goddard Space Flight Center, EOSDIS Core System (ECS) Contract Data Requirements Document
430-TP-004-001	Distributed Database Architecture of Data Management Subsystem
430-TP-003-001	DBMS Benchmark Report
515-CD-001-003	Availability Models/Predictions for the ECS Project
516-CD-001-003	Reliability Predictions for the ECS Project
518-CD-001-003	Maintainability Predictions for the ECS Project
IMSV0-PD-SD-002 v1.0.11 950515	Messages and Development Data Dictionary for v4.5 IMS Client

2.3 Information Documents Not Referenced

The following documents, although not referenced herein and/or not directly applicable, do amplify and clarify the information presented in this document. These documents are not binding on the content of the SDPS Design Specifications.

205-CD-002-001	Science User's Guide and Operations Procedure Handbook for the ECS Project. Part 4: Software Developer's Guide to Preparation, Delivery, Integration, and Test with ECS
206-CD-001-002	Version 0 Analysis Report for the ECS Project
209-CD-010-001	Interface Control Document Between EOSDIS Core System (ECS) and the Langley Research Center (LaRC) Distributed Active Archive Center (DAAC) Draft
194-302-DV2-001	ECS Facilities Plan for the ECS Project
101-303-DV1-001	Individual Facility Requirements for the ECS Project, Preliminary
194-317-DV1-001	Prototyping and Studies Plan for the ECS Project
318-CD-003-XXX	Prototyping and Studies Progress Report for the ECS Project (monthly)
333-CD-002-003	SDP Toolkit Users Guide for the ECS Project
601-CD-001-002	Maintenance and Operations Management Plan for the ECS Project
194-604-OP1-001	ECS Operations Concept Document for the ECS Project, Working Draft
101-620-OP2-001	List of Recommended Maintenance Equipment for the ECS Project
194-703-PP1-001	System Design Review (SDR) Presentation Package for the ECS Project
193-801-SD4-001	PGS Toolkit Requirements Specification for the ECS Project
194-813-SI4-002	Planning and Scheduling Prototype Results Report for the ECS Project
194-813-SI4-003	DADS Prototype One FSMS Product Operational Evaluation
194-813-SI4-004	DADS Prototype One STK Wolfcreek 9360 Automated Cartridge System Hardware Characterization Report
813-RD-009-001	DADS Prototype Two Multi-FSMS Product Integration Evaluation

828-RD-001-002	Government Furnished Property for the ECS Project
193-WP-118-001	Algorithm Integration and Test Issues for the ECS Project
193-WP-611-001	Science-based System Architecture Drivers for the ECS Project, Revision 1.0
193-WP-623-001	ECS Evolutionary Development White Paper
194-WP-901-002	EOSDIS Core System Science Information Architecture, White Paper, Working Paper
194-WP-902-002	ECS Science Requirements Summary, White Paper, Working Paper
194-WP-904-002	Multi-Track Development for the ECS Project, White Paper, Working Paper
194-WP-913-003	User Environment Definition for the ECS Project, White Paper, Working Paper
194-WP-914-001	CORBA Object Request Broker Survey for the ECS Project, White Paper, Working Paper
194-WP-918-001	DADS Prototype One FSMS Product Operational Evaluation, White Paper, Draft Report
194-WP-925-001	Science Software Integration and Test, White Paper, Working Paper
420-WP-001-001	Maximizing the Use of COTS Software in the SDPS SDS Software Design, White Paper
193-TP-626-001	GCDIS/UserDIS Study ECS Technical Paper, Draft 0.2
194-TP-266-002	Data Distribution Architecture Logical Object Model (LOM) for the ECS Project, Version 2.01
194-TP-267-001	Data Server Architecture Logical Object Model (LOM) for the ECS Project, Version 2.00
194-TP-313-001	ECS User Characterization Methodology and Results
194-TP-316-001	Data Compression Study for the ECS Project
194-TP-548-001	User Scenario Functional Analysis [for the ECS Project]
194-TP-569-001	PDPS Prototyping at ECS Science and Technology Laboratory, Progress Report #4
222-TP-003-005	Release Plan Content Description for the ECS Project
430-TP-001-001	SDP Toolkit Implementation with Pathfinder SSM/I Precipitation Rate Algorithm, Technical Paper
440-TP-001-001	Science Data Server Architecture Study [for the ECS Project]
420-TD-001-001	ECS Data Server Taxonomy Technical Description
-TR-	Hughes Training, Inc., ECS User Interface Style Guide, White Paper, Version 5.0
423-16-01	Goddard Space Flight Center, Data Production Software and Science Computing Facility (SCF) Standards and Guidelines

423-41-02	Goddard Space Flight Center, Functional and Performance Requirements Specification for the Earth Observing System Data and Information System (EOSDIS) Core System
540-022	Goddard Space Flight Center, Earth Observing System (EOS) Communications (Ecom) System Design Specification
560-EDOS-0211.0001	Goddard Space Flight Center, Interface Requirements Document Between EDOS and the EOS Ground System (EGS)

3. Release A SDPS Data Management Subsystem Overview

3.1 Subsystem Overview

The Release A SDPS Data Management Subsystem provides services which search for, locate and access data on behalf of a user or another program. Data management services decouple users and programs from the methods used by a site to access the data, and the manner in which the data have been named. The Release A SDPS Data Management Subsystem consists of only one CSCI, the V0 Interoperability Services (GTWAY) which provides interface between ECS data server and the Version 0 client. The Data management hardware CI (DMGHW) provides the hardware support needed by Release A SDPS Data Management Subsystem and Interoperability subsystem software components.

3.2 Subsystem Structure

The Release A SDPS Data Management Subsystem is composed of one CSCI and one HWCI:

- V0 Gateway (GTWAY) is a software component. It provides interoperability services between ECS data server and V0 client.
- Data Management HWCI (DMGHW) is a hardware component. It provides hardware for both Data Management subsystem and Interoperability subsystem CSCIs.

Figure 3.2-1 illustrates the Release A SDPS Data Management Subsystem context within ECS. For each of the context diagram flows, Table 3.2-1 contains a description of the subsystem interfaces.

In the table, where an exact number is unavailable, the data volume is estimated as low (less than 1 MB), medium (between 1 MB and 1 GB), or high (greater than 1 GB) per use defined in the frequency column. The frequency information will be updated as the interfaces are fully defined.

3.3 Subsystem Design Rationale

The following are the drivers for the design of the Release A SDPS Data Management Subsystem:

- a. Decouple users and programs from the methods used by sites to access the data, and the manner in which the data are named.
- b. Provide interoperability between V0 client and ECS data server.
- c. Simplify data administration
- d. Provide site autonomy

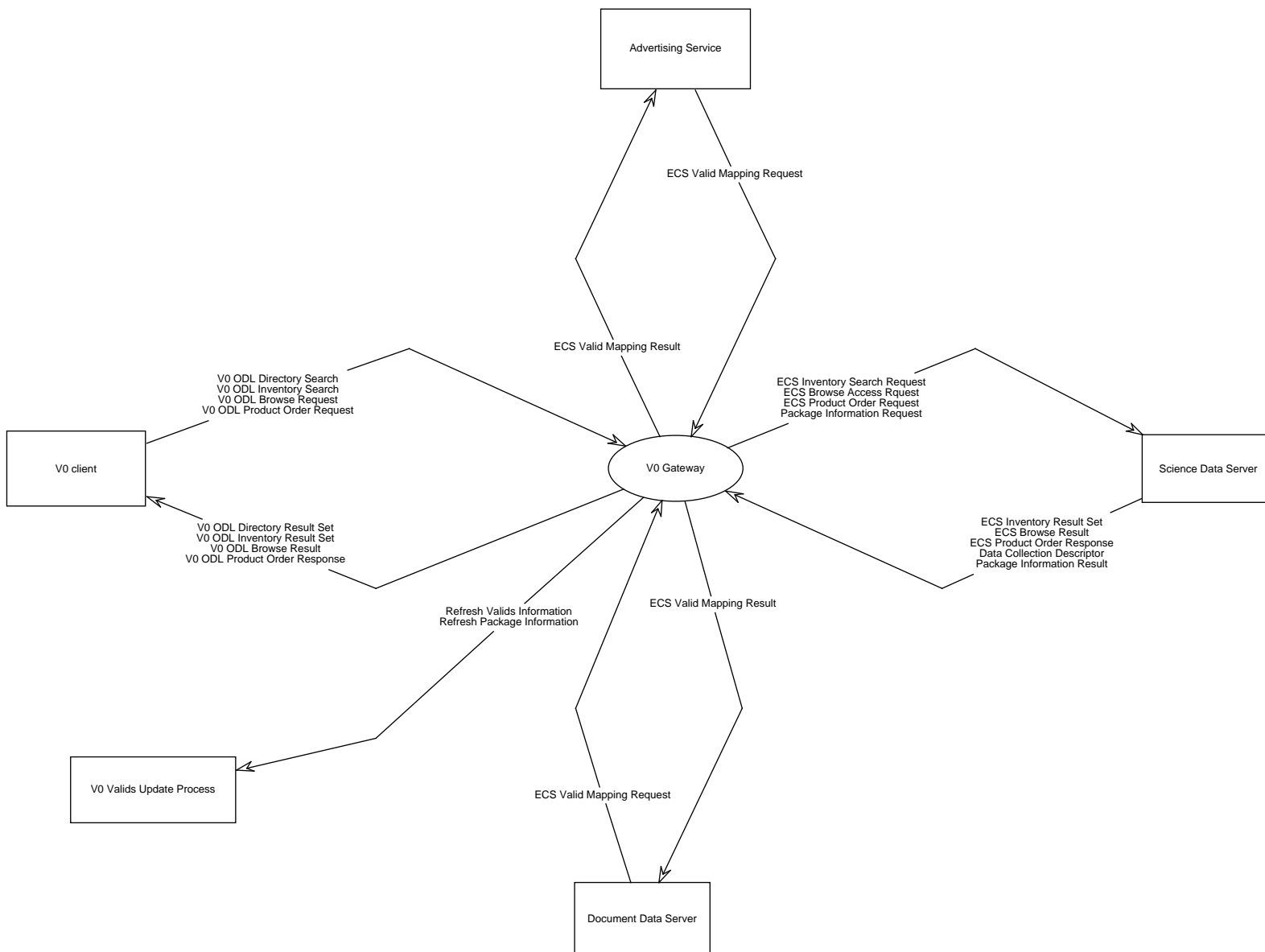


Figure 3.2-1. Data Management Subsystem Context

Table 3.2-1. Subsystem Interfaces (1 of 2)

Source	Destination	Data Types	Data Volume	Frequency
V0 Client	V0 Gateway	V0 Directory Search Requests	low	as requested
V0 Gateway	V0 Client	V0 Directory Search Result Set	low	in response to request
V0 Client	V0 Gateway	V0 Inventory Search Requests	low	as requested
V0 Gateway	Science Data Server	ECS Inventory Search Requests	low	in response to a V0 inventory search request
Science Data Server	V0 Gateway	ECS Inventory Result Set	low-high	in response to ECS inventory search request
V0 Gateway	V0 Client	V0 Inventory Result Set	low-high	in response to ECS inventory result Set
V0 Client	V0 Gateway	V0 Browse Request	low	as requested
V0 Gateway	Science Data Server	ECS Browse Request	low	in response to V0 Browse Request
Science Data Server	V0 Gateway	ECS Browse Result	low-medium	in response to ECS Browse Request
V0 Gateway	V0 Client	V0 Browse Result	low-medium	in response to ECS Browse Result
V0 Client	V0 Gateway	V0 Product Order Request	low	frequency dependent on user input
V0 Gateway	Science Data Server	ECS Product Order Request	low	in response to V0 product request
Science Data Server	V0 Gateway	ECS Product Order Response	low	in response to ECS Product Request
V0 Gateway	V0 Client	V0 Product Order Response	low	in response to ECS Product Request Response
Science Data Server	V0 Gateway	Data Collection Descriptor	low	whenever new ESDT is created
Document Data Server	V0 Gateway	ECS Valid Mapping Request	low	as requested

Table 3.2-1. Subsystem Interfaces (2 of 2)

Source	Destination	Data Types	Data Volume	Frequency
V0 Gateway	Document Data Server	ECS Valid Mapping Result	low	in response to the ECS Valid Mapping request
Advertising Service	V0 Gateway	ECS Valid Mapping Request	low	as requested
V0 Gateway	Advertising Service	ECS Valid Mapping Result	low	in response to the ECS Valid Mapping request
V0 Gateway	Science Data Server	Package Information Request	low	as requested by the administrator
Science Data Server	V0 Gateway	Package Information Result	low	in response to the package information request
V0 Gateway	V0 Valids Update Process	Refresh Package Information	low	in response to the Package Information Result from the data server
V0 Gateway	V0 Valids Update Process	Refresh Valids Information	low	in response to Data Server Schema Export

4. GTWAY - Version 0 Gateway CSCI

4.1 CSCI Overview

Gateway provides interoperability with V0 for directory queries, inventory queries, browse requests and product orders. Version 0 queries originating from Version 0 clients will be sent to a Version 0 gateway which will operate at each DAAC. The gateway will translate an incoming V0 ODL request into ECS query format and submit it to the local ECS data server. The result will be returned to the Gateway, which then will reformat it into V0 ODL structures and return it to V0 client. The structure of the V0 ODL messages is documented in "Messages and Development Data Dictionary for v4.5 of IMS Client" (IMSV0-PD-SD-002 v1.0.11 950515). The Gateway uses a database constructed by a gateway administrator using the V0 search parameters, ECS schema and metadata. The Advertising Service (ADSRV) CSCI and Document Data Server (DDSRV) CSCI make use of the gateway database to resolve ECS to V0 mappings.

Since this CSCI is on the incremental development track, requirements, schedule, scenarios, issues and design are documented in a Software Development File (SDF) for GTWAY.

4.2 CSCI Context

GTWAY CSCI is the only CSCI in the Release A SDPS Data Management Subsystem for Release A. Therefore the context of the CSCI is identical to the subsystem context which is shown in Figure 3.2-1.

4.3 CSCI Object Model

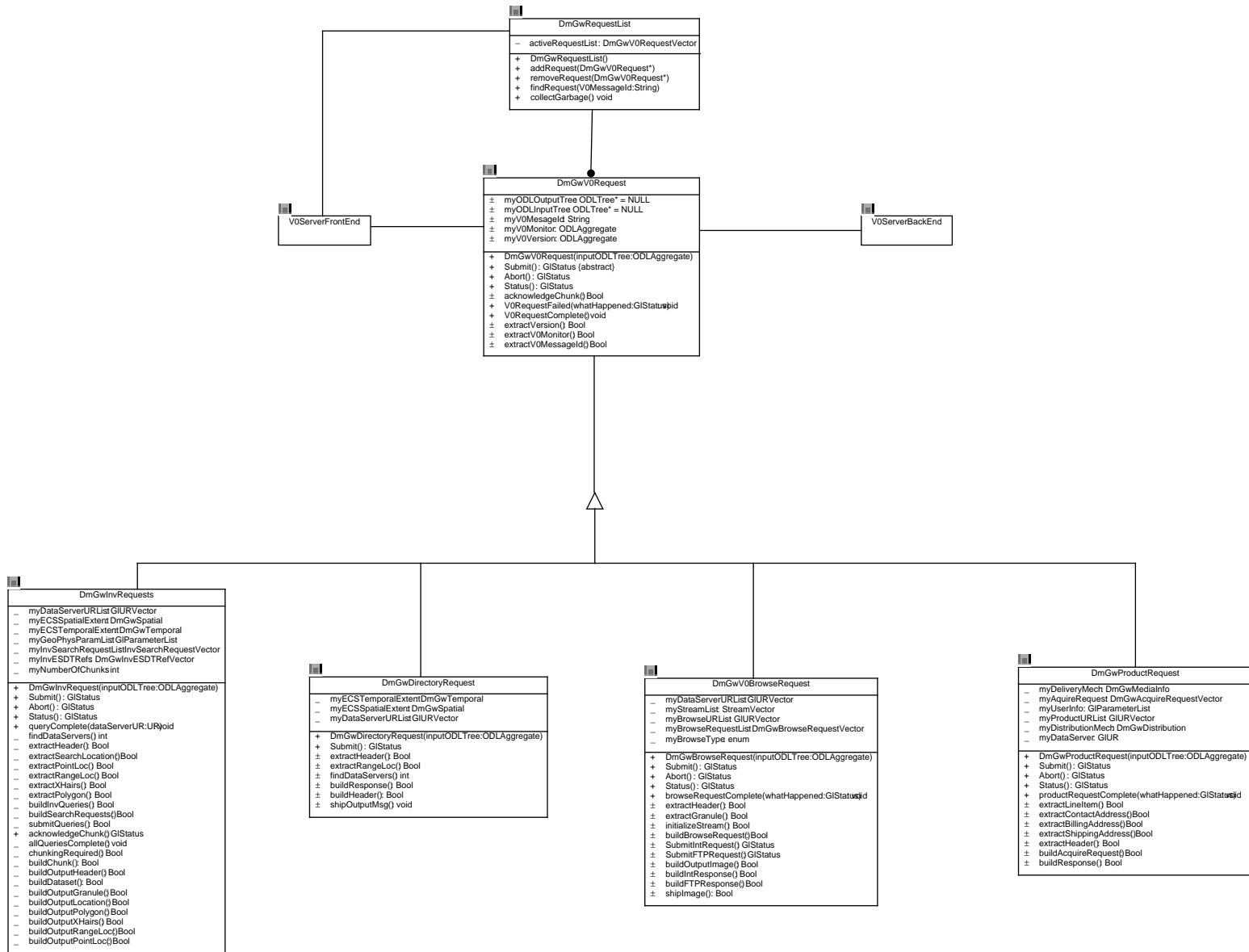
The GTWAY CSCI has been separated into four class categories:

- Request Processing
- Data Server Interface
- Persistent Data
- V0-ECS Mapper

Each of these class categories is described in the following subsections.

4.3.1 GTWAY Request Processing Object Model

This object model (Figure 4.3.1-1) represents the classes containing the requests serviced by the GTWAY CSCI and the corresponding result classes. Each V0 request is converted into an ECS request and is submitted to the data server using data server interface classes described in Section 4.3.2. The result set received from the data server is translated into V0 ODL results message. Requests can be of four types: directory search, inventory search, browse request and product request. A directory search is resolved using the gateway persistent classes described in Section 4.3.3. The result set for directory search contains the GCMD entry ID, the data collection name and center where it originated from, all of which are stored in the gateway persistent objects. An inventory search request is resolved partially by the gateway using the valids stored in the gateway persistent objects i.e., the gateway identifies the data collection names from the given geophysical param-



4.3.1-1. DMGW-Requests Object Model Diagram

ters, field campaign names, sensor names and satellite names. For each data collection identified, a separate request is formulated to submit to the data server. Data Server sends the results set for each data collection back to the gateway and the gateway formulates the ODL response from the ECS result set. Gateway receives the browse request with the granule id. Gateway persistent objects keep track of the mapping between the Universal Reference (UR) and the granule ID of each granule that was returned to the V0 Client for a certain period. The granule id is translated to a Universal Reference (UR) before submitting the request to the data server. If a browse product is available for direct viewing on the users desktop, the data server returns the browse product with the result set, or it sends a response that the request is being processed for FTP. Product Order requests are translated into ECS data distribution requests and submitted to the data server. The notification received from the data server is translated by the gateway to an ODL response and sent to the V0 client.

4.3.1.1 DmGwDirectoryRequest Class

Parent Class: DmGwV0Request

Public: No Distributed Object: No

Purpose and Description:

The DmGwDirectoryRequest class is a specialization of the DmGwV0Request class. This class is responsible for processing a V0 Directory Search request.

Attributes:

myDataServerURLList - stores the list of data server GIURs that are returned from the directory search.

Data Type: GIURVector

Privilege: Private

Default Value:

myECSSpatialExtent - stores the spatial extent to be used in the directory search.

Data Type: DmGwSpatial

Privilege: Private

Default Value:

myECSTemporalExtent - stores the temporal range to be used in the directory search.

Data Type: DmGwTemporal

Privilege: Private

Default Value:

Operations:

DmGwDirectoryRequest - The DmGwDirectoryRequest constructor initializes the myECSTemporalExtent and myECSSpatialExtent attributes from information contained in myODLInputTree.

Arguments: inputODLTree:ODLAggregate

Return Type: Void

Privilege: Public

Submit - The Submit member function is a specialization of the DmGwV0Request base class member function. It extracts the search constraints from myODLInputTree, performs a directory search on the gateway database, and formats the search output into myODLOutputTree.

Arguments:

Return Type: GIStatus

Privilege: Public

buildHeader - The buildHeader member function builds the output header in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Protected

buildResponse - The buildResponse member functions inserts the directory search results into myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Protected

extractHeader - The extractHeader member function initializes myECSTemporalExtent attribute from information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

extractRangeLoc - The extractRangeLoc member function initializes the myECSSpatialExtent attribute from the RangeLoc information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

findDataServers - The findDataServers member function queries the gateway database to perform the inventory search.

Arguments:

Return Type: int

Privilege: Protected

shipOutputMsg - The shipOutputMsg member function delivers myODLOutputTree to the V0ServerBackEnd.

Arguments:

Return Type: void

Privilege: Protected

Associations:

The DmGwDirectoryRequest class has associations with the following classes:
None

4.3.1.2 DmGwInvRequests Class

Parent Class: DmGwV0Request

Public: No Distributed Object: No

Purpose and Description:

The DmGwInvRequest class is a specialization of the DmGwV0Request class. This class is responsible for processing a V0 Inventory Search Request.

Attributes:

myDataServerURLList - list of GIUR pointers for the data servers to be searched.

Data Type: GIURVector

Privilege: Private

Default Value:

myECSSpatialExtent - spatial constraint specified for the inventory search.

Data Type: DmGwSpatial

Privilege: Private

Default Value:

myECSTemporalExtent - temporal constraint specified for the inventory search.

Data Type: DmGwTemporal

Privilege: Private

Default Value:

myGeoPhysParamList - list of geophysical parameters specified for the inventory search.

Data Type: GIParameterList

Privilege: Private

Default Value:

myInvESDTRefs - list of DmGwInvESDRReferences that are returned as a result of a data server search.

Data Type: DmGwInvESDTRefVector

Privilege: Private

Default Value:

myInvSearchRequestList - list of DmGwInvSearchRequest pointers. There is one pointer per each data server to be searched.

Data Type: InvSearchRequestVector

Privilege: Private

Default Value:

myNumberOfChunks - stores the number of V0 Chunk responses required.

Data Type: int

Privilege: Private

Default Value:

Operations:

Abort - The Abort member function is a specialization of the DmGwV0Request base class member function. It aborts any outstanding data server search requests and terminates further processing of this inventory search.

Arguments:

Return Type: GIStatus

Privilege: Public

DmGwInvRequest - The DmGwInvRequest constructor.

Arguments: inputODLTree:ODLAggregate

Return Type: Void

Privilege: Public

Status - The Status member function is a specialization of the DmGwV0Request base class member function. It returns the current status of the inventory search being processed.

Arguments:

Return Type: GIStatus

Privilege: Public

Submit - The Submit member function is a specialization of the DmGwV0Request base class member function. It extracts the search constraints from myODLInputTree, determines which data servers should be queried, builds search request for each data server, and submits the requests to the data servers.

Arguments:

Return Type: GIStatus

Privilege: Public

acknowledgeChunk - The acknowledgeChunk member function is a specialization of the DmGwV0Request base class member function. It prepares and ships the next response chunk.

Arguments:

Return Type: GIStatus

Privilege: Public

allQueriesComplete - The allQueriesComplete member function prepares the ODL response message in myODLOutputTree. If chunking is required is builds and ships the first chunk only.

Arguments:

Return Type: void

Privilege: Private

buildChunk - The buildChunk member function builds the myODLOutputTree attribute for one chunk of inventory results.

Arguments:

Return Type: Bool

Privilege: Private

buildDataset - The buildDataset member function builds the dataset header in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Private

buildInvQueries - The buildInvQueries member function builds a DmGwInvQuery object for each data server to be queried.

Arguments:

Return Type: Bool

Privilege: Private

buildOutputGranule - The buildOutputGranule builds the granule information in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Private

buildOutputHeader - The buildOutputHeader builds the response header in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Private

buildOutputLocation - The buildOutputLocation builds the granule spatial location in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Private

buildOutputPointLoc - The buildOutputPointLoc builds the granule PointLoc spatial location in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Private

buildOutputPolygon - The buildOutputPolygon builds the granule Polygon spatial location in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Private

buildOutputRangeLoc - The buildOutputLocation builds the granule RangeLoc spatial location in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Private

buildOutputXHairs - The buildOutputLocation builds the granule spatial location in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Private

buildSearchRequests - The buildInvQueries member function builds a DmGwInvSearchRequest object for each data server to be queried.

Arguments:

Return Type: Bool

Privilege: Private

chunkingRequired - The chunkingRequired member function returns True if myODLOutputTree is greater than 64K.

Arguments:

Return Type: Bool

Privilege: Private

extractHeader - The extractHeader member function initializes the myGeoPhysParamList and myECSTemporalExtent attributes from information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Private

extractPointLoc - The extractPointLoc member function initializes the myECSSpatialExtent attribute from the pointLoc information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Private

extractPolygon - The extractPolygon member function initializes the myECSSpatialExtent attribute from the polygon information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Private

extractRangeLoc - The extractRangeLoc member function initializes the myECSSpatialExtent attribute from the RangeLoc information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Private

extractSearchLocation - The extractSearchLocation member function determines the type of spatial information that is contained in myODLInputTree. Based on the spatial type, the appropriate extract member function will be called.

Arguments:

Return Type: Bool

Privilege: Private

extractXHairs - The extractXHairs member function initializes the myECSSpatialExtent attribute from the XHairs information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Private

findDataServers - The findDataServers member function queries the gateway database to determine which data servers should be queried for the inventory search.

Arguments:

Return Type: int

Privilege: Private

queryComplete - The queryComplete member function is called by each DmGwInvSearchRequest object when its data server search has completed.

Arguments: dataServerUR:UR

Return Type: void

Privilege: Public

submitQueries - The submitQueries member function submits each of the DmGwInvSearchRequest objects.

Arguments:

Return Type: Bool

Privilege: Private

Associations:

The DmGwInvRequests class has associations with the following classes:

None

4.3.1.3 DmGwProductRequest Class

Parent Class: DmGwV0Request

Public: No Distributed Object: No

Purpose and Description:

The DmGwProductRequestClass is a specialization of the DmGwV0Request class. This class is responsible for processing a V0 Product Request (i.e. data order request).

Attributes:

myAcquireRequest - stores the DmGwAcquireRequest for the product order.

Data Type: DmGwAcquireRequestVector

Privilege: Private

Default Value:

myDataServer - stores the UR of the data server to be used for the product request.

Data Type: GIUR

Privilege: Private

Default Value:

myDeliveryMech - stores the DmGwMediaInfo type for the product request.

Data Type: DmGwMediaInfo

Privilege: Private

Default Value:

myDistributionMech - stores the data server distribution interface object.

Data Type: DmGwDistribution

Privilege: Private

Default Value:

myProductURLList - list of URs for the data items to be ordered.

Data Type: GIURVector

Privilege: Private

Default Value:

myUserInfo - stores the list of user information (e.g. e-mail address) needed for a product order.

Data Type: GIParameterList

Privilege: Private

Default Value:

Operations:

Abort - The Status member function is a specialization of the DmGwV0Request base class member function. It aborts any outstanding distribution requests and terminates further processing of this product request.

Arguments:

Return Type: GIStatus

Privilege: Public

DmGwProductRequest - The DmGwProductRequest constructor.

Arguments: inputODLTree:ODLAggregate

Return Type: Void

Privilege: Public

Status - The Status member function is a specialization of the DmGwV0Request base class member function. It returns the current status of the inventory search being processed.

Arguments:

Return Type: GIStatus

Privilege: Public

Submit - The Submit member function is a specialization of the DmGwV0Request base class member function. It uses myDistributionMech to order the product data.

Arguments:

Return Type: GIStatus

Privilege: Public

buildAcquireRequest - The buildAcquireRequest member function initializes the myAcquireRequest attribute.

Arguments:

Return Type: Bool

Privilege: Protected

buildResponse - The buildResponse member function is responsible for building the V0 response message in myODLOutputTree.

Arguments:

Return Type: Bool

Privilege: Protected

extractBillingAddress - The extractBillingAddress member function initializes a portion of myUserInfo from myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

extractContactAddress - The extractContactAddress member function initializes portions of myUserInfo from myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

extractHeader - The extractHeader member function controls the initialization of class attributes from information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

extractLineItem - The extractLineItem member function extracts the granule order data from the LineItem portion of myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

extractShippingAddress - The extractShippingAddress member function initializes a portion of myUserInfo from myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

productRequestComplete - The productRequestComplete member function is called by the DmGwAcquireRequest object when the acquire operation is complete.

Arguments: whatHappened:GIStatus

Return Type: void

Privilege: Public

Associations:

The DmGwProductRequest class has associations with the following classes:

None

4.3.1.4 DmGwRequestList Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Purpose and Description:

The DmGwRequestList class is a container object that maintains the list of currently active V0 Requests. Each V0 Request is represented by a DmGwV0Request object.

Attributes:

activeRequestList - stores the collection of active DmGwV0Request objects.

Data Type: DmGwV0RequestVector

Privilege: Private

Default Value:

Operations:

DmGwRequestList - The DmGwRequestList constructor initializes the activeRequestList.

Arguments:

Return Type: Void

Privilege: Public

addRequest - The addRequest member function adds a new DmGwV0Request object to the activeRequestList.

Arguments: DmGwV0Request*

Return Type: Void

Privilege: Public

collectGarbage - The collectGarbage member function is responsible for the removal and deletion of DmGwV0Request objects. Whenever a DmGwV0Request object has completed processing, it registers itself for garbage collection by calling the removeRequest member function of the DmGwRequestList object. The request list object then schedules a garbage collection callback which initiates the collectGarbage member function.

Arguments:

Return Type: void

Privilege: Public

findRequest - The findRequest member function returns the address of a DmGwV0Request object with the V0MessageId specified in the argument. An object pointer will only be returned if it is currently on the activeREquestList.

Arguments: V0MessageId:String

Return Type: Void

Privilege: Public

removeRequest - The removeRequest member function removes a DmGwV0Request from the activeRequestList and schedules the deletion of the DmGwV0Request via the collectGarbage member function.

Arguments: DmGwV0Request*

Return Type: Void

Privilege: Public

Associations:

The DmGwRequestList class has associations with the following classes:

Class: DmGwV0Request

Class: V0ServerFrontEnd

4.3.1.5 DmGwV0BrowseRequest Class

Parent Class: DmGwV0Request

Public: No Distributed Object: No

Purpose and Description:

The DmGwV0BrowseRequest is a specialization of the DmGwV0Request class. This class is responsible for processing A V0 Browse request.

Attributes:

myBrowseRequestList - stores the list of DmGwBrowseRequest objects that are processing the individual browse requests.

Data Type: DmGwBrowseRequestVector

Privilege: Private

Default Value:

myBrowseType - stores the type of browse acquisition. Possible types are integrated and FTP-Pull.

Data Type: enum

Privilege: Private

Default Value:

myBrowseURLList - stores the list of GIURs which point to the browse data to be obtained.

Data Type: GIURVector

Privilege: Private

Default Value:

myDataServerURLList - stores the list of GIURs for the data servers which will process the browse requests.

Data Type: GIURVector

Privilege: Private

Default Value:

myStreamList - stores the list of stream objects which will receive the browse data from the data server.

Data Type: StreamVector

Privilege: Private

Default Value:

Operations:

Abort - The Abort member function is a specialization of the DmGwV0Request base class member function. It aborts any outstanding DmGwBrowseRequest objects in myBrowseRequestList and terminates further processing of this V0 Browse Request.

Arguments:

Return Type: GIStatus

Privilege: Public

DmGwBrowseRequest - The DmGwBrowseRequest constructor initializes the class attribute set from information contained in myODLInputTree.

Arguments: inputODLTree:ODLAggregate

Return Type: Void

Privilege: Public

Status - The Status member function is a specialization of the DmGwV0Request base class member function. It returns the current status of the browse request being processed.

Arguments:

Return Type: GIStatus

Privilege: Public

Submit - The Submit member function is a specialization of the DmGwV0Request base class member function. It submits the browse request by invoking either SubmitIntRequest or SubmitFTPRequest.

Arguments:

Return Type: GIStatus

Privilege: Public

SubmitFTPRequest - The SubmitFtpRequest submits an FTP-Pull browse request to the data server via myBrowseRequestList.

Arguments:

Return Type: GIStatus

Privilege: Protected

SubmitIntRequest - The SubmitIntRequest submits an integrated browse request to the data server via myBrowseRequestList.

Arguments:

Return Type: GIStatus

Privilege: Protected

browseRequestComplete - The browseRequestComplete member function is called by each of the DmGwBroseRequest objects in myBrowseRequestList as they complete the browse request to the data server.

Arguments: whatHappened:GIStatus

Return Type: void

Privilege: Public

buildBrowseRequest - The buildBrowseRequest member function initializes each object in myBrowseRequestList.

Arguments:

Return Type: Bool

Privilege: Protected

buildFTPResponse - The buildFTPResponse member function builds the myODLOutputTree attribute for an FTP-Pull browse response message.

Arguments:

Return Type: Bool

Privilege: Protected

buildIntResponse - The buildIntResponse member function builds the myODLOutputTree attribute for an integrated browse response message.

Arguments:

Return Type: Bool

Privilege: Protected

buildOutputImage - The buildOutputImage member function delivers the image data from myStreamList to the V0ServerBackEnd object.

Arguments:

Return Type: Bool

Privilege: Protected

extractGranule - the extractGranule member function initializes the myBroseURList from information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

extractHeader - The extractHeader member function initializes the myBrowseType attribute from information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

initializeStream - The initializeStream member function initializes each of the stream objects in myStreamList.

Arguments:

Return Type: Bool

Privilege: Protected

shipImage - The shipImage member function delivers the myODLOutputTree attribute to the V0ServerBackEnd for transmission to the V0 Client.

Arguments:

Return Type: Bool

Privilege: Protected

Associations:

The DmGwV0BrowseRequest class has associations with the following classes:

None

4.3.1.6 DmGwV0Request Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Purpose and Description:

The DmGwV0Request object is the base class from which specialized versions (e.g. DmGwV0BrowseRequest) are derived. DmGwV0Request abstracts the operations and attributes that are common to all gateway V0 request processing. DmGwV0Request objects are created by the V0ServerFrontEnd component.

Attributes:

myODLInputTree - stores the internal representation of the ODL message received from the V0 Client.

Data Type: ODLTree*

Privilege: Protected

Default Value: NULL

myODLOutputTree - stores the internal representation of the ODL response message to be delivered to the V0 Client.

Data Type: ODLTree*

Privilege: Protected

Default Value: NULL

myV0MesageId - stores the V0 Message Id from myODLInputTree.

Data Type: String

Privilege: Protected

Default Value:

myV0Monitor - stores the V0 Monitor information from myODLInputTree.

Data Type: ODLAggregate

Privilege: Protected

Default Value:

myV0Version - stores the V0 Version information from myODLInputTree.

Data Type: ODLAggregate

Privilege: Protected

Default Value:

Operations:

Abort - The Abort member function stops all further processing of the DmGwV0Request object.

Arguments:

Return Type: GlStatus

Privilege: Public

DmGwV0Request - The DmGwV0Request constructor initializes the myODLOutputTree attribute from the inputODLTree argument and extracts the common V0 ODL attributes. Upon completion, the DmGwV0Request object is ready to begin processing the request.

Arguments: inputODLTree:ODLAggregate

Return Type: Void

Privilege: Public

Status - The Status member function returns the current status of the DmGwV0Request object via a GlStatus object.

Arguments:

Return Type: GlStatus

Privilege: Public

Submit - The Submit member function initiates processing of the DmGwV0Request class. After the object has been constructed, the Submit operation begins the actual request processing.

Arguments:

Return Type: GlStatus

Privilege: Public

This is an abstract operation

V0RequestComplete - The V0RequestComplete member function schedules the object for garbage collection.

Arguments:

Return Type: void

Privilege: Public

V0RequestFailed - The V0RequestFailed member function stops further processing, and schedules the DmGwV0Request object for garbage collection.

Arguments: whatHappened:GlStatus

Return Type: void

Privilege: Public

acknowledgeChunk - The acknowledgeChunk member function is called by the V0ServerFrontEnd code whenever a previously delivered chunk is acknowledged by the V0 Client.

Arguments:

Return Type: Bool

Privilege: Protected

extractV0MessageId - The extractV0MessageId initializes the myV0MessageId attribute from information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

extractV0Monitor - The extractV0Monitor member function extracts the V0 Monitor information from myODLInputTree and initializes the myV0Monitor attribute.

Arguments:

Return Type: Bool

Privilege: Protected

extractVersion - The extractVersion member function initializes the myV0Version attribute from information contained in myODLInputTree.

Arguments:

Return Type: Bool

Privilege: Protected

Associations:

The DmGwV0Request class has associations with the following classes:

Class: DmGwRequestList

Class: V0ServerBackEnd

Class: V0ServerFrontEnd

4.3.1.7 V0ServerBackEnd Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Purpose and Description:

The V0ServerBackEnd is an OTS package provided by the V0 IMS Server.

Attributes:

None

Operations:

None

Associations:

The V0ServerBackEnd class has associations with the following classes:

Class: DmGwV0Request

4.3.1.8 V0ServerFrontEnd Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Purpose and Description:

The V0ServerFrontEnd is an OTS package provided by the V0 IMS Server.

Attributes:

None

Operations:

None

Associations:

The V0ServerFrontEnd class has associations with the following classes:

Class: DmGwRequestList

Class: DmGwV0Request

4.3.2 GTWAY Data Server Interface Object Model

This object model (Figure 4.3.2-1) represents the interface classes for the Gateway Subsystem to the Data Server Subsystem. For each service request directed from the Request Processing Model, this model provides a service interface. It consists of the objects for handling Inventory Search Requests, Browse Requests, Acquisition Requests, Valid and Distribution Media Requests. The interface objects communicate with the designated data server to submit the service requests. When the service requests are completed, the integrated result and status information are returned back to the Request Processing Module for further processing.

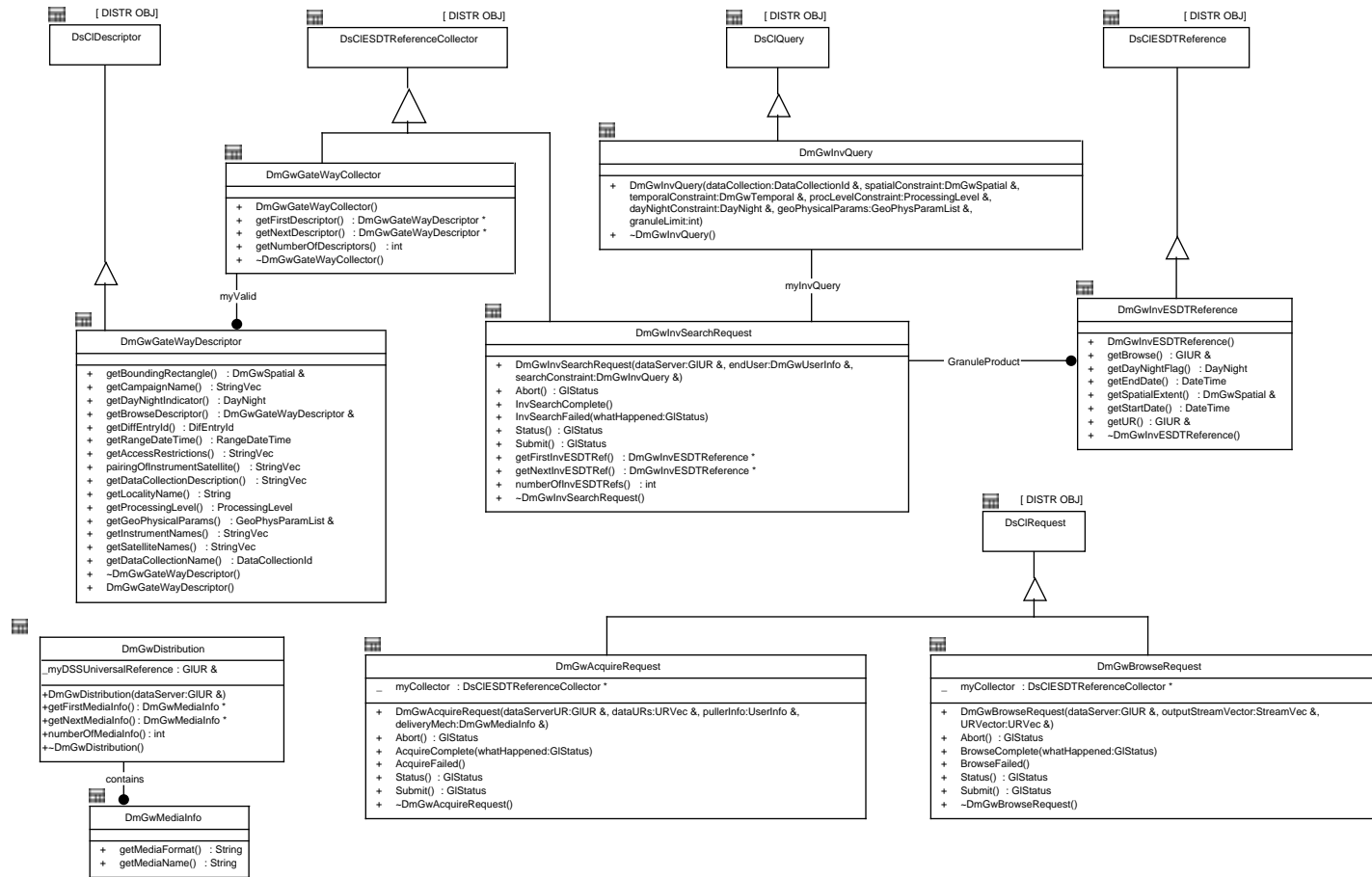


Figure 4.3.2-1. DMGWDataServerIF Object Model Diagram

4.3.2.1 DmGwAcquireRequest Class

Parent Class: DsClRequest

Public: Yes Distributed Object: No

Purpose and Description:

This class contains all the information and the operations required to submit a product ordering request to the data server.

Attributes:

myCollector - This attribute contains the universal reference to the data server which the data collection being requested is binded to.

Data Type: DsClESDTReferenceCollector *

Privilege: Private

Default Value:

Operations:

Abort - This operation cancels the product acquisition request that is being submitted.

Arguments:

Return Type: GIStatus

Privilege: Public

AcquireComplete - This operation should be invoked by the communication layer after the product acquisition request submitted to the data server is completed successfully.

Arguments: whatHappened:GIStatus

Return Type: Void

Privilege: Public

AcquireFailed - This operation should be invoked by the communication layer if a product acquisition request submitted to the data server can not be completed due to the reason specified in the argument whatHappened.

Arguments:

Return Type: Void

Privilege: Public

DmGwAcquireRequest - This operation constructs a product ordering request.

Arguments: dataServerUR:GIUR &, dataURs:URVec &, pullerInfo:UserInfo &, deliveryMech:DmGwMediaInfo &

Return Type: Void

Privilege: Public

Status - This operation is invoked if a desire to check the status of product acquisition is required.

Arguments:

Return Type: GIStatus

Privilege: Public

Submit - This operation is invoked to submit the product acquisition request to the data server.

Arguments:

Return Type: GIStatus

Privilege: Public

~DmGwAcquireRequest - This operation destroys the structure of product ordering request.

Arguments:

Return Type: Void

Privilege: Public

Associations:

The DmGwAcquireRequest class has associations with the following classes:

None

4.3.2.2 DmGwBrowseRequest Class

Parent Class: DsCIRequest

Public: Yes Distributed Object: No

Purpose and Description:

This class contains all the information and the operations required to submit a browse request to the data server.

Attributes:

myCollector - This attribute contains the universal reference to the data server which the browse image being requested is binded to.

Data Type: DsCIESDTRreferenceCollector *

Privilege: Private

Default Value:

Operations:

Abort - This operation cancels the browse image request that is being submitted.

Arguments:

Return Type: GIStatus

Privilege: Public

BrowseComplete - This operation should be invoked by the communication layer after a browse request submitted to the data server is completed successfully.

Arguments: whatHappened:GIStatus

Return Type: Void

Privilege: Public

BrowseFailed - This operation should be invoked by the communication layer if a browse request submitted to the data server can not be completed due to the reason specified in the argument whatHappened.

Arguments:
Return Type: Void
Privilege: Public

DmGwBrowseRequest - This operation constructs a browse image request.
Arguments: dataServer:GIUR &, outputStreamVector:StreamVec &, URVector:URVec &
Return Type: Void
Privilege: Public

Status - This operation is invoked if a desire to check the status of browse request is required.
Arguments:
Return Type: GIStatus
Privilege: Public

Submit - This operation is invoked to submit the browse image request to the data server.
Arguments:
Return Type: GIStatus
Privilege: Public

~DmGwBrowseRequest - This operation destroys the structure of browse image request.
Arguments:
Return Type: Void
Privilege: Public

Associations:

The DmGwBrowseRequest class has associations with the following classes:
None

4.3.2.3 DmGwDistribution Class

Parent Class: Not Applicable
Public: Yes Distributed Object: No
Purpose and Description:
This class contains all the information and the operations required to acquire product distribution and format information.

Attributes:

myDSSUniversalReference - This attribute contains the universal reference to the distribution data server.
Data Type: GIUR &
Privilege: Private
Default Value:

Operations:

DmGwDistribution - This operation constructs a distribution request for the product format information to the distribution data server.

Arguments: dataServer:GIUR &

Return Type: Void

Privilege: Public

getFirstMediaInfo - This operation retrieves the information of first media format available for the product being ordered.

Arguments:

Return Type: DmGwMediaInfo *

Privilege: Public

getNextMediaInfo - This operation retrieves the information of next media format available for the product being ordered.

Arguments:

Return Type: DmGwMediaInfo *

Privilege: Public

numberOfMediaInfo - This operation retrieves the number of different media formats for the product being ordered.

Arguments:

Return Type: int

Privilege: Public

~DmGwDistribution - This operation destroys the structure of the distribution request for the product format information.

Arguments:

Return Type: Void

Privilege: Public

Associations:

The DmGwDistribution class has associations with the following classes:

Class: DmGwMediaInfo contains - This association indicates that the DmGwDistribution contains a collection of DmGwMediaInfo.

4.3.2.4 DmGwGateWayCollector Class

Parent Class: DsCIESDTRreferenceCollector

Public: Yes Distributed Object: No

Purpose and Description:

This class contains all the information and the operations required to retrieve a collection of valids exported from the data server.

Attributes:

All Attributes inherited from parent class

Operations:

DmGwGateWayCollector - This operation constructs a valid export request to the data server.

Arguments:

Return Type: Void

Privilege: Public

getFirstDescriptor - This operation retrieves the information of the first valid descriptor exported from the data server.

Arguments:

Return Type: DmGwGateWayDescriptor *

Privilege: Public

getNextDescriptor - This operation retrieves the information of the next valid descriptor exported from the data server.

Arguments:

Return Type: DmGwGateWayDescriptor *

Privilege: Public

getNumberOfDescriptors - This operation retrieves the number of valid descriptors exported from the data server.

Arguments:

Return Type: int

Privilege: Public

~DmGwGateWayCollector - This operation destroys the structure of the valid export request.

Arguments:

Return Type: Void

Privilege: Public

Associations:

The DmGwGateWayCollector class has associations with the following classes:

Class: DmGwGateWayDescriptor myValid - This association indicates that DmGwGateWayCollector is a collection of DmGwGateWayDescriptor valids.

4.3.2.5 DmGwGateWayDescriptor Class

Parent Class: DsClDescriptor

Public: Yes Distributed Object: No

Purpose and Description:

This class contains the valid information which is returned by the data server after the valid export request is completed.

Attributes:

All Attributes inherited from parent class

Operations:

DmGwGatewayDescriptor - This operation constructs a valid descriptor for storing exported valid information.

Arguments:

Return Type: Void

Privilege: Public

getAccessRestrictions - This operation returns the ordering restriction and legal prerequisites placed on the data set.

Arguments:

Return Type: StringVec

Privilege: Public

getBoundingRectangle - This operation returns the specification of the spatial coverage for each data set.

Arguments:

Return Type: DmGwSpatial &

Privilege: Public

getBrowseDescriptor - This operation returns the browse descriptor, if exists, which is related to the data set.

Arguments:

Return Type: DmGwGatewayDescriptor &

Privilege: Public

getCampaignName - This operation returns the name of campaign or project that gathered the data set.

Arguments:

Return Type: StringVec

Privilege: Public

getDataCollectionDescription - This operation returns the major emphasis of the content of the data collection.

Arguments:

Return Type: StringVec

Privilege: Public

getDataCollectionName - This operation returns the recommended name to be used when referring to this data set.

Arguments:

Return Type: DataCollectionId

Privilege: Public

getDayNightIndicator - This operation returns the day/night indicator for the data set.

Arguments:

Return Type: DayNight

Privilege: Public

getDiffEntryId - This operation returns the entry id of the Global Change Master Directory for the data set.

Arguments:

Return Type: DifEntryId

Privilege: Public

getGeoPhysicalParams - This operation returns the specification of the geophysical parameters referenced in the data set.

Arguments:

Return Type: GeoPhysParamList &

Privilege: Public

getInstrumentNames - This operation returns the abbreviation, acronym, or other common name of the instrument sensor by which the data set is collected.

Arguments:

Return Type: StringVec

Privilege: Public

getLocalityName - This operation returns the spacial coverage described for the data set.

Arguments:

Return Type: String

Privilege: Public

getProcessingLevel - This operation returns the classification of the science data processing level which defines the characteristics of the data set.

Arguments:

Return Type: ProcessingLevel

Privilege: Public

getRangeDateTime - This operation returns the temporal coverage period extended for the data set.

Arguments:

Return Type: RangeDateTime

Privilege: Public

getSatelliteNames - This operation returns the name of the satellite on which the data set is collected.

Arguments:

Return Type: StringVec

Privilege: Public

pairingOfInstrumentSatellite - This operation returns the matching pairs for instrument sensor and satellite.

Arguments:

Return Type: StringVec

Privilege: Public

~DmGwGateWayDescriptor - This operation destroys the structure of valid descriptor.

Arguments:

Return Type: Void

Privilege: Public

Associations:

The DmGwGateWayDescriptor class has associations with the following classes:

Class: DmGwGateWayCollector myValid - This association indicates that DmGwGateWayCollector is a collection of DmGwGateWayDescriptor valids.

4.3.2.6 DmGwInvESDTReference Class

Parent Class: DsCIESDTReference

Public: Yes Distributed Object: No

Purpose and Description:

This class contains the information of granule references returned from an inventory search request for a particular dataset.

Attributes:

All Attributes inherited from parent class

Operations:

DmGwInvESDTReference - This operation constructs the structure for storing granule related information.

Arguments:

Return Type: Void

Privilege: Public

getBrowse - This operation retrieves the browse reference, if exists, which is related to the individual granule returned in the inventory data set.

Arguments:

Return Type: GIUR &

Privilege: Public

getDayNightFlag - This operation retrieves the day/night indication of individual granule returned for the inventory data set.

Arguments:

Return Type: DayNight

Privilege: Public

getEndDate - This operation retrieves the ending time of the temporal coverage for the individual granule returned in the inventory data set.

Arguments:

Return Type: DateTime

Privilege: Public

getSpatialExtent - This operation retrieves the spatial coverage of individual granule returned in the inventory data set.

Arguments:

Return Type: DmGwSpatial &

Privilege: Public

getStartDate - This operation retrieves the starting time of the temporal coverage for the individual granule returned in the inventory data set.

Arguments:

Return Type: DateTime

Privilege: Public

getUR - This operation returns the universal reference to the individual granule returned in the inventory data set.

Arguments:

Return Type: GIUR &

Privilege: Public

~DmGwInvESDTRreference - This operation destroys the granule reference structure.

Arguments:

Return Type: Void

Privilege: Public

Associations:

The DmGwInvESDTRreference class has associations with the following classes:

Class: DmGwInvSearchRequest GranuleProduct - This association indicates that the DmGwInvESDTRreference is a collection of granule product returned from the inventory search request DmGwInvSearchRequest.

4.3.2.7 DmGwInvQuery Class

Parent Class: DsCIQuery

Public: Yes Distributed Object: No

Purpose and Description:

This class contains the information of the query criteria for the data set requested.

Attributes:

All Attributes inherited from parent class

Operations:

DmGwInvQuery - This class contains the information of the query criteria for the data set requested.

Arguments: dataCollection:DataCollectionId &, spatialConstraint:DmGwSpatial &, temporalConstraint:DmGwTemporal &, procLevelConstraint:ProcessingLevel &, dayNightConstraint:DayNight &, geoPhysicalParams:GeoPhysParamList &, granuleLimit:int
Return Type: Void
Privilege: Public

~DmGwInvQuery - This operation destroys the query structure for the data set.
Arguments:
Return Type: Void
Privilege: Public

Associations:

The DmGwInvQuery class has associations with the following classes:

Class: DmGwInvSearchRequest myInvQuery - This association indicates that DmGwInvQuery is the query criteria when the inventory search request DmGwInvSearchRequest is submitted.

4.3.2.8 DmGwInvSearchRequest Class

Parent Class: DsCIESDTRreferenceCollector

Public: Yes Distributed Object: No

Purpose and Description:

This class contains all the information and the operations required to submit an inventory search request to the data server.

Attributes:

All Attributes inherited from parent class

Operations:

Abort - This operation cancels the inventory search request that is being submitted.

Arguments:
Return Type: GIStatus
Privilege: Public

DmGwInvSearchRequest - This operation constructs an inventory search request.

Arguments: dataServer:GIUR &, endUser:DmGwUserInfo &, searchConstraint:DmGwInvQuery &
Return Type: Void
Privilege: Public

InvSearchComplete - This operation should be invoked by the communication layer after the inventory search request submitted to the data server is completed successfully.

Arguments:
Return Type: Void
Privilege: Public

InvSearchFailed - This operation should be invoked by the communication layer if an inventory search request submitted to the data server can not be completed due to the reason specified in the argument whatHappened.

Arguments: whatHappened:GIStatus

Return Type: Void

Privilege: Public

Status - This operation is invoked if a desire to check the status of inventory search request is required.

Arguments:

Return Type: GIStatus

Privilege: Public

Submit - This operation is invoked to submit the inventory search request to the data server.

Arguments:

Return Type: GIStatus

Privilege: Public

getFirstInvESDTRef - This operation retrieves the information of the first granule reference returned from the inventory search request.

Arguments:

Return Type: DmGwInvESDTReference *

Privilege: Public

getNextInvESDTRef - This operation retrieves the information of the next granule reference returned from an inventory search request.

Arguments:

Return Type: DmGwInvESDTReference *

Privilege: Public

numberOfInvESDTRefs - This operation retrieves the number of granule references returned in the data set requested.

Arguments:

Return Type: int

Privilege: Public

~DmGwInvSearchRequest - This operation destroys the structure of inventory search request.

Arguments:

Return Type: Void

Privilege: Public

Associations:

The DmGwInvSearchRequest class has associations with the following classes:

Class: DmGwInvESDTRreference GranuleProduct - This association indicates that the DmGwInvESDTRreference is a collection of granule product returned from the inventory search request DmGwInvSearchRequest.

Class: DmGwInvQuery myInvQuery - This association indicates that DmGwInvQuery is the query criteria when the inventory search request DmGwInvSearchRequest is submitted.

4.3.2.9 DmGwMediaInfo Class

Parent Class: Not Applicable

Public: Yes Distributed Object: No

Purpose and Description:

This class contains all the information and the operations required for acquiring distribution media format.

Attributes:

None

Operations:

getMediaFormat - This operation retrieves the format of the distribution media.

Arguments:

Return Type: String

Privilege: Public

getMediaName - This operation retrieves the name of the distribution media.

Arguments:

Return Type: String

Privilege: Public

Associations:

The DmGwMediaInfo class has associations with the following classes:

Class: DmGwDistribution contains - This association indicates that the DmGwDistribution contains a collection of DmGwMediaInfo.

4.3.2.10 DsClDescriptor Class

Parent Class: Not Applicable

Public: Yes Distributed Object: Yes

Purpose and Description:

The DsClDescriptor class is the proxy class which is imported from the Data Server Subsystem. The description of the server class is defined in the DID305 Data Server Subsystem section.

Attributes:

None

Operations:

None

Associations:

The DsCIDescriptor class has associations with the following classes:

None

4.3.2.11 DsCIESDTReference Class

Parent Class: Not Applicable

Public: Yes Distributed Object: Yes

Purpose and Description:

The DsCIESDTReference class is the proxy class which is imported from the Data Server Subsystem. The description of the server class is defined in the DID305 Data Server Subsystem section.

Attributes:

None

Operations:

None

Associations:

The DsCIESDTReference class has associations with the following classes:

None

4.3.2.12 DsCIESDTReferenceCollector Class

Parent Class: Not Applicable

Public: Yes Distributed Object: Yes

Purpose and Description:

The DsCIESDTReferenceCollector class is the proxy class which is imported from the Data Server Subsystem. The description of the server class is defined in the DID305 Data Server Subsystem section.

Attributes:

None

Operations:

None

Associations:

The DsCIESDTReferenceCollector class has associations with the following classes:

None

4.3.2.13 DsCIQuery Class

Parent Class: Not Applicable

Public: Yes Distributed Object: Yes

Purpose and Description:

The DsClQuery class is the proxy class which is imported from the Data Server Subsystem. The description of the server class is defined in the DID305 Data Server Subsystem section.

Attributes:

None

Operations:

None

Associations:

The DsClQuery class has associations with the following classes:

None

4.3.2.14 DsClRequest Class

Parent Class: Not Applicable

Public: Yes Distributed Object: Yes

Purpose and Description:

The DsClESDTRrequest class is the proxy class which is imported from the Data Server Subsystem. The description of the server class is defined in the DID305 Data Server Subsystem section.

Attributes:

None

Operations:

None

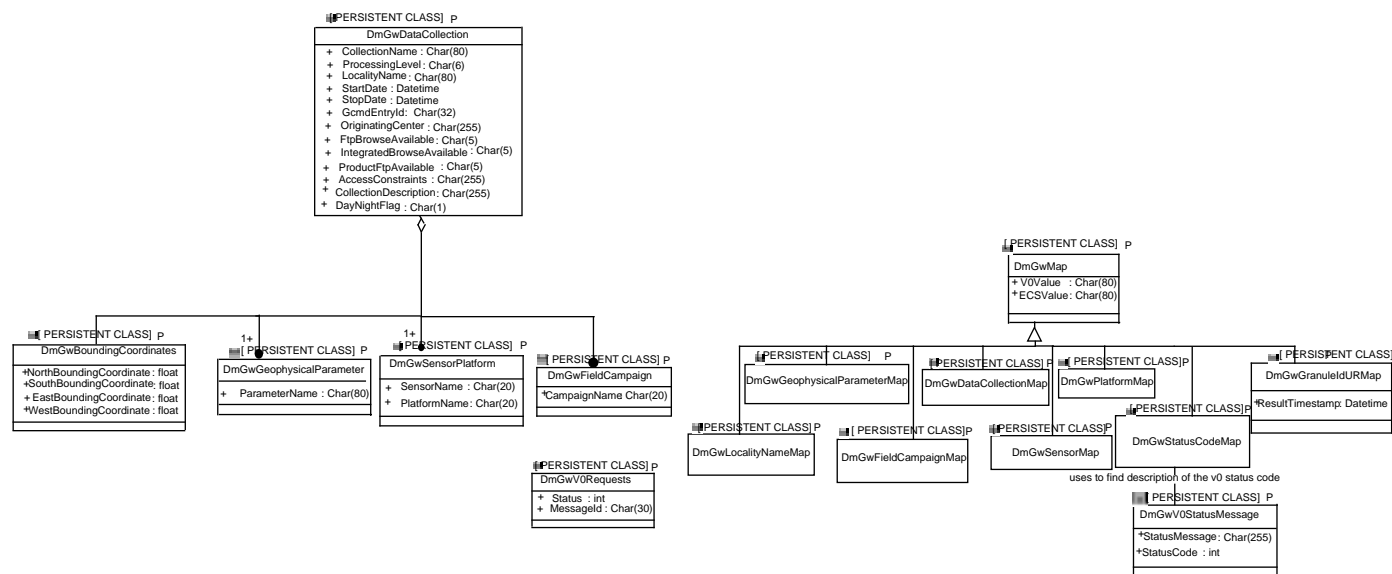
Associations:

The DsClRequest class has associations with the following classes:

None

4.3.3 GTWAY Persistent Data Object Model

V0 Gateway database stores three types of information: Valid, V0-ECS mapping tables and request tracking. This object model (Figure 4.3.3-1) includes all the above. Whenever a new ESDT is created by the data server it exports that data type and its valid information. This export information, after being mapped using V0 ECS mapping service is stored in valid persistent module i.e., DmGwDataCollection in its part classes. When the V0 gateway request processing module receives a request for directory search, it resolves the query, using DmGwDataCollection and its part classes. All the required information for resolving the query is stored in the local database. The V0 gateway request processing module resolves the inventory search messages partly using the valid persistent module and determines the list of data collection names to send to the ECS data server. It uses the V0 ECS mapping service to change any of the terms that are being sent to the data server. When the inventory results are returned from the ECS data server, V0 gateway mapping service keeps track of the granule id and the UR relationship using DmGwGranuleIdURMap class. This data is mainly used when the client requests the browse product or tries to order a granule using the granule id. These mappings will be available in the database for a certain period. V0 ECS mapping service maps the ECS terms to V0 terms and vice versa using the contents of the subclasses



4.3.3-1. DMGWPersistentData Object Model Diagram

of DmGwMap. DmGwV0Requests class is intended to be used for recovering the state in case of failures. Current V0 client is not built to recover to the previous state, but this capability is included to meet the requirements in case that should change in future.

4.3.3.1 DmGwBoundingCoordinates Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class is a part class of DmGwDataCollection. It contains the bounding coordinates of the spatial coverage of the data collection.

Attributes:

EastBoundingCoordinate - Easternmost longitude of the data collection spatial coverage.

Data Type: float

Privilege: Public

Default Value:

NorthBoundingCoordinate - Northernmost latitude of the data collection spatial coverage

Data Type: float

Privilege: Public

Default Value:

SouthBoundingCoordinate - Southernmost latitude of the data collection spatial coverage

Data Type: float

Privilege: Public

Default Value:

WestBoundingCoordinate - Westernmost longitude of the data collection spatial coverage.

Data Type: float

Privilege: Public

Default Value:

Operations:

None

Associations:

The DmGwBoundingCoordinates class has associations with the following classes:

DmGwDataCollection (Aggregation)

4.3.3.2 DmGwDataCollection Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores the metadata about the data collection stored in the data server associated with V0 gateway. The information in this class and its part classes is used to resolve a V0 directory query and to build the V0 valids file.

Attributes:

AccessConstraints - Descriptive notes about any access restrictions on the data collection.

Data Type: Char(255)

Privilege: Public

Default Value:

CollectionDescription - Brief description of the data collection

Data Type: Char(255)

Privilege: Public

Default Value:

CollectionName - Name of the data collection which is same as V0 DATASET_ID

Data Type: Char(80)

Privilege: Public

Default Value:

DayNightFlag - This flag indicates whether or not the data collection is completely either day or night

Data Type: Char(1)

Privilege: Public

Default Value:

FtpBrowseAvailable - It indicates if browse product for the data collection is available through FTP or not. It is same as V0 BROWSE, FTP attribute

Data Type: Char(5)

Privilege: Public

Default Value:

GcmdEntryId - The id assigned by the Global Change Master Directory (GCMD) for the data collection. It is equivalent to V0 attribute MD_ENTRY_ID.

Data Type: Char(32)

Privilege: Public

Default Value:

IntegratedBrowseAvailable - It indicates if the browse product can be viewed through the client. It is same as V0 BROWSE, INTEGRATED attribute.

Data Type: Char(5)

Privilege: Public

Default Value:

LocalityName - This attribute provides a name denoting the spatial coverage of the data collection. It is equivalent to V0 DATASET_COVERAGE, SPATIAL.

Data Type: Char(80)

Privilege: Public

Default Value:

OriginatingCenter - The data center from where DIF for the data collection has originated. This attribute is used by V0 for GCMD search. It is equivalent to the V0 ORG_CENTER attribute.

Data Type: Char(255)

Privilege: Public

Default Value:

ProcessingLevel - This attribute reflects the classification of the science data processing level, which defines in general terms the characteristics of the output of the processing performed. This is equivalent to V0 PROCESSING_LEVEL attribute.

Data Type: Char(6)

Privilege: Public

Default Value:

ProductFtpAvailable - It indicates whether or not the data collection is available through FTP. It is same as V0 FTP_PRODUCT_AVAILABLE.

Data Type: Char(5)

Privilege: Public

Default Value:

StartDate - This attribute is the beginning date and time of the data collection temporal coverage.

Data Type: Datetime

Privilege: Public

Default Value:

StopDate - It defines the ending date and time of the data collection temporal coverage.

Data Type: Datetime

Privilege: Public

Default Value:

Operations:

None

Associations:

The DmGwDataCollection class has associations with the following classes:
None

4.3.3.3 DmGwDataCollectionMap Class

Parent Class: DmGwMap

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores the mappings, if any, between the dataset names of V0 clients and ECS data server.

Attributes:

All Attributes inherited from parent class

Operations:

All Operations inherited from parent class

Associations:

The DmGwDataCollectionMap class has associations with the following classes:
None

4.3.3.4 DmGwFieldCampaign Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This is a part class of DmGwDataCollection. A data collection may be collected using a campaign which is contained in this class. It is used in resolving the directory search from V0 client and validating the inventory search from V0 client.

Attributes:

CampaignName - Name of the campaign the data collection is associated with.

Data Type: Char(20)

Privilege: Public

Default Value:

Operations:

None

Associations:

The DmGwFieldCampaign class has associations with the following classes:
DmGwDataCollection (Aggregation)

4.3.3.5 DmGwFieldCampaignMap Class

Parent Class: DmGwMap

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores the mapping between V0 field campaign names and the ECS field campaign names.

Attributes:

All Attributes inherited from parent class

Operations:

All Operations inherited from parent class

Associations:

The DmGwFieldCampaignMap class has associations with the following classes:

None

4.3.3.6 DmGwGeophysicalParameter Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This is a part class of DmGwDataCollection. It contains the geophysical parameter names that the data collection references.

Attributes:

ParameterName - Name of the geophysical parameter that the data collection references.

Data Type: Char(80)

Privilege: Public

Default Value:

Operations:

None

Associations:

The DmGwGeophysicalParameter class has associations with the following classes:

DmGwDataCollection (Aggregation)

4.3.3.7 DmGwGeophysicalParameterMap Class

Parent Class: DmGwMap

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores the mappings between V0 geophysical parameters and ECS geophysical parameters. A single V0 parameter may map to many ECS parameters and vice versa.

Attributes:

All Attributes inherited from parent class

Operations:

All Operations inherited from parent class

Associations:

The DmGwGeophysicalParameterMap class has associations with the following classes:

None

4.3.3.8 DmGwGranuleIdURMap Class

Parent Class: DmGwMap

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores the UR and the granule id of every result granule of an inventory search. This mapping is maintained in the V0 gateway database so that any further requests from the V0 client using the granule id can be translated into UR before requesting the data server. These results are stored for certain period of time and then purged out.

Attributes:

ResultTimestamp - Timestamp when the result was returned to the V0 client. It is used in purging the mapping after certain period of time.

Data Type: Datetime

Privilege: Public

Default Value:

Operations:

All Operations inherited from parent class

Associations:

The DmGwGranuleIdURMap class has associations with the following classes:

None

4.3.3.9 DmGwLocalityNameMap Class

Parent Class: DmGwMap

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores mapping from V0 locality names (which are mapped to DATASET_COVERAGE, SPATIAL) to ECS locality names.

Attributes:

All Attributes inherited from parent class

Operations:

All Operations inherited from parent class

Associations:

The DmGwLocalityNameMap class has associations with the following classes:
None

4.3.3.10 DmGwMap Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class provides the mappings between the V0 terms and the ECS terms. V0 valids such as geophysical parameters, sensors, platforms may have different terminology in ECS for which the mappings are stored using this class. This is an abstract class where term can be of several types and they appear as subclasses of this class.

Attributes:

ECSValue - ECS term

Data Type: Char(80)

Privilege: Public

Default Value:

V0Value - V0 term

Data Type: Char(80)

Privilege: Public

Default Value:

Operations:

None

Associations:

The DmGwMap class has associations with the following classes:
None

4.3.3.11 DmGwPlatformMap Class

Parent Class: DmGwMap

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores the mappings between V0 platforms and the ECS platforms.

Attributes:

All Attributes inherited from parent class

Operations:

All Operations inherited from parent class

Associations:

The DmGwPlatformMap class has associations with the following classes:
None

4.3.3.12 DmGwSensorMap Class

Parent Class: DmGwMap

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores the mappings between the V0 sensor names and ECS sensor names

Attributes:

All Attributes inherited from parent class

Operations:

All Operations inherited from parent class

Associations:

The DmGwSensorMap class has associations with the following classes:
None

4.3.3.13 DmGwSensorPlatform Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This is a part class of the DmGwDataCollection. It contains the sensor names used for measurement of the data collection and its associated platform names. This class is used in resolving V0 directory search requests and validating V0 inventory queries.

Attributes:

PlatformName - Name of the platform where the sensor used for the data collection was housed.

Data Type: Char(20)

Privilege: Public

Default Value:

SensorName - Name of the sensor or instrument used for collecting the data collection.

Data Type: Char(20)

Privilege: Public

Default Value:

Operations:

None

Associations:

The DmGwSensorPlatform class has associations with the following classes:
DmGwDataCollection (Aggregation)

4.3.3.14 DmGwStatusCodeMap Class

Parent Class: DmGwMap

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores the mappings between ECS status codes and V0 status codes. Many ECS status codes may map to a single V0 status code and vice versa

Attributes:

All Attributes inherited from parent class

Operations:

All Operations inherited from parent class

Associations:

The DmGwStatusCodeMap class has associations with the following classes:
Class: DmGwV0StatusMessage usestofinddescriptionofthev0statuscode

4.3.3.15 DmGwV0Requests Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class keeps track of the V0 requests coming in and their status to perform housekeeping.

Attributes:

MessageId - V0 client generated message id.

Data Type: Char(30)

Privilege: Public

Default Value:

Status - Current status of the message

Data Type: int

Privilege: Public

Default Value:

Operations:

None

Associations:

The DmGwV0Requests class has associations with the following classes:
None

4.3.3.16 DmGwV0StatusMessage Class

Parent Class: Not Applicable

Public: No Distributed Object: No

Persistent Class: True

Purpose and Description:

This class stores the list of V0 status codes and the associated messages.

Attributes:

StatusCode - V0 Status code

Data Type: int

Privilege: Public

Default Value:

StatusMessage - Descriptive message what the status code means

Data Type: Char(255)

Privilege: Public

Default Value:

Operations:

None

Associations:

The DmGwV0StatusMessage class has associations with the following classes:

Class: DmGwStatusCodeMap usestofinddescriptionofthev0statuscode

4.3.4 GTWAY V0 ECS Mapping Service Object Model

V0 ECS Mapping Service provides the mappings between V0 and ECS schema and the valids. This object model (Figure 4.3.4-1) represents the classes and operations required to perform the necessary mapping. Some V0 attributes need to be converted into the ECS domain using a function. For example, a polygon search area specification or a temporal range specification needs conversion to ECS spatial extent specification and temporal extent specification. Certain attributes such as geophysical parameters need domain mapping between the V0 terms and the ECS terms. These domain mappings are available to the mapping service through the gateway persistent objects. When the data server performs the schema export, gateway uses the mapping service to map the ECS terms to V0 terms

4.3.4.1 DmGwV0ECSMapper Class

Parent Class: Not Applicable

Public: Yes Distributed Object: Yes

Purpose and Description:

This class contains all the information and the operations required to map the V0 attributes to ECS and to map the ECS attributes to V0.

Attributes:

None

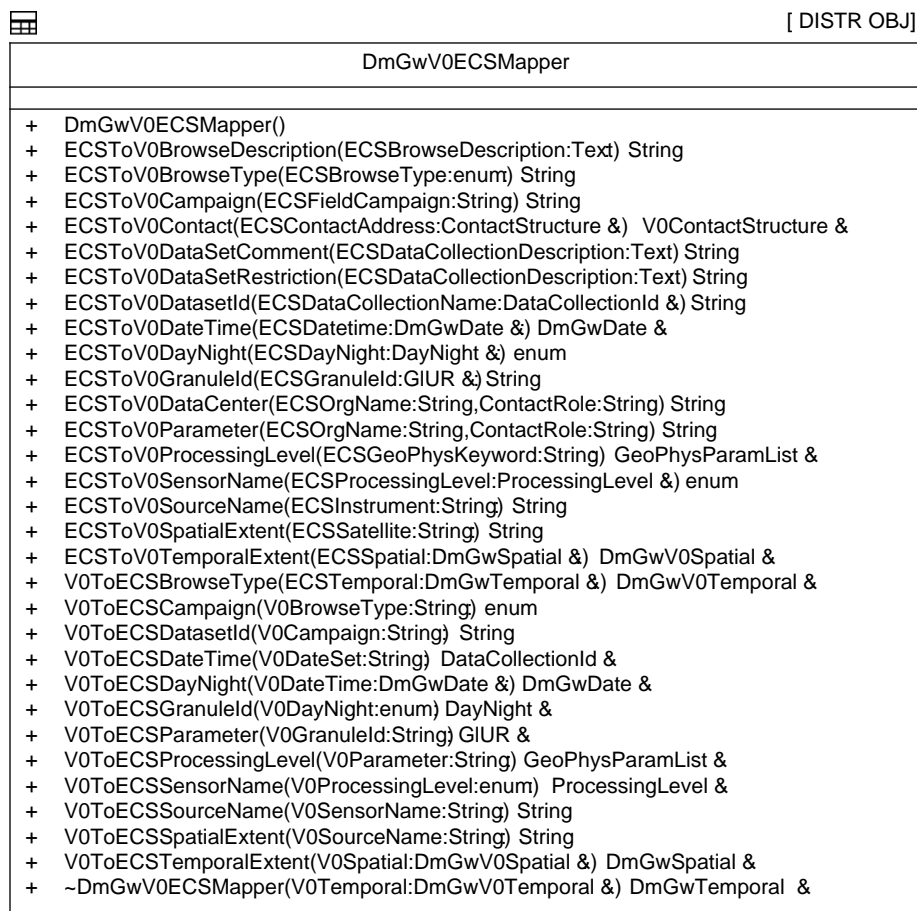


Figure 4.3.4-1. DMGWV0ECSMapper Object Model Diagram

Operations:

DmGwV0ECSMapper - This operation constructs the Mapper class which performs the mapping from V0 to ECS and vice versa.

Arguments:

Return Type: Void

Privilege: Public

ECSToV0BrowseDescription - This operation maps the ECS browse image description to the V0 browse description format for a specific data collection.

Arguments: ECSEBrowseDescription:Text
Return Type: String
Privilege: Public

ECSToV0BrowseType - This operation maps the ECS type of delivery for the browse image to the V0 type.

Arguments: ECSEBrowseType:enum
Return Type: String
Privilege: Public

ECSToV0Campaign - This operation maps the ECS name of campaign or project that gathered the dataset to the V0 type.

Arguments: ECSEFieldCampaign:String
Return Type: String
Privilege: Public

ECSToV0Contact - This operation maps the ECS point of contact reference for a data collection or browse image to the V0 format of contact reference.

Arguments: ECSEContactAddress:ContactStructure &
Return Type: V0ContactStructure &
Privilege: Public

ECSToV0DataCenter - This operation maps the acronym of the data center which transmits the data information to the V0 format.

Arguments: ECSEOrgName:String,ContactRole:String
Return Type: String
Privilege: Public

ECSToV0DataSetComment - This operation maps the ECS description for a particular data collection to the V0 format of comment information.

Arguments: ECSEDataCollectionDescription:Text
Return Type: String
Privilege: Public

ECSToV0DataSetRestriction - This operation maps the ECS ordering restriction and legal prerequisites placed on the data collection to the V0 restriction format.

Arguments: ECSEDataCollectionDescription:Text
Return Type: String
Privilege: Public

ECSToV0DatasetId - This operation maps the ECS name which identifies the data collection and associated granules to the V0 name.

Arguments: ECSEDataCollectionName:DataCollectionId &
Return Type: String
Privilege: Public

ECSToV0DateTime - This operation maps a particular ECS date and time format to the V0 data and time format.

Arguments: ECSDatetime:DmGwDate &

Return Type: DmGwDate &

Privilege: Public

ECSToV0DayNight - This operation maps the ECS day/night indication to the V0 day/night indication format.

Arguments: ECSDayNight:DayNight &

Return Type: enum

Privilege: Public

ECSToV0GranuleId - This operation translates the ECS universal reference to the individual granule in the data collection to the V0 format of granule id.

Arguments: ECSGranuleId:GIUR &

Return Type: String

Privilege: Public

ECSToV0Parameter

Arguments: ECSOrgName:String,ContactRole:String

Return Type: String

Privilege: Public

ECSToV0ProcessingLevel

Arguments: ECSGeoPhysKeyword:String

Return Type: GeoPhysParamList &

Privilege: Public

ECSToV0SensorName

Arguments: ECSProcessingLevel:ProcessingLevel &

Return Type: enum

Privilege: Public

ECSToV0SourceName

Arguments: ECSInstrument:String

Return Type: String

Privilege: Public

ECSToV0SpatialExtent

Arguments: ECSSatellite:String

Return Type: String

Privilege: Public

ECSToV0TemporalExtent

Arguments: ECSSpatial:DmGwSpatial &
Return Type: DmGwV0Spatial &
Privilege: Public

V0ToECSBrowseType

Arguments: ECSTemporal:DmGwTemporal &
Return Type: DmGwV0Temporal &
Privilege: Public

V0ToECSCampaign

Arguments: V0BrowseType:String
Return Type: enum
Privilege: Public

V0ToECSDatasetId

Arguments: V0Campaign:String
Return Type: String
Privilege: Public

V0ToECSDateTime

Arguments: V0DateSet:String
Return Type: DataCollectionId &
Privilege: Public

V0ToECSDayNight

Arguments: V0DateTime:DmGwDate &
Return Type: DmGwDate &
Privilege: Public

V0ToECSGranuleId

Arguments: V0DayNight:enum
Return Type: DayNight &
Privilege: Public

V0ToECSParameter

Arguments: V0GranuleId:String
Return Type: GIUR &
Privilege: Public

V0ToECSProcessingLevel

Arguments: V0Parameter:String
Return Type: GeoPhysParamList &
Privilege: Public

V0ToECSSensorName

Arguments: V0ProcessingLevel:enum

Return Type: ProcessingLevel &

Privilege: Public

V0ToECSSourceName

Arguments: V0SensorName:String

Return Type: String

Privilege: Public

V0ToECSSpatialExtent

Arguments: V0SourceName:String

Return Type: String

Privilege: Public

V0ToECSTemporalExtent

Arguments: V0Spatial:DmGwV0Spatial &

Return Type: DmGwSpatial &

Privilege: Public

~DmGwV0ECSEMapper

Arguments: V0Temporal:DmGwV0Temporal &

Return Type: DmGwTemporal &

Privilege: Public

Associations:

The DmGwV0ECSEMapper class has associations with the following classes:

None

4.4 GTWAY - Version 0 Gateway CSCI Structure

Table 4.4-1 provides a summary of the components which make up this CSCI. Since this CSCI is on an incremental development track, the table presents the current best estimate of the CSCI components. These components are likely to change as the CSCI evolves.

Table 4.4-1. Gateway CSCI Components

Name	Description	Type (DEV or OTS)
Gateway server	Application server that processes the gateway requests	DEV
V0 IMS server	System level V0 IMS server libraries	OTS
Gateway DBMS	DBMS used to store data server schema and some meta-data information.	OTS

5. DMGHW - Data Management HWCI

The Data Management HWCI (DMGHW) is the primary HWCI within CIDM. The DMGHW CI provides the necessary hardware resources for the persistent storage of dictionary and schema data. The DMGHW CI will be sized to support the service demands of the Advertising CI and Gateway CI services in Release A, and will expand to include the Data Dictionary CI, Local Information Manager CI and Distributed Information Manager CI services in Release B. The primary technologies that apply to the DMGHW CI include DBMS servers, World Wide Web servers, host attached disk, possible use of RAID disk and a variety of communications capabilities. A small pool of local operations workstations will support DBMS management, data repository administration, data specialist, user support and phone/mail support functions. The operations workstations are few in number and small in scale. The scope of DBMS management, data repository administration, data specialist, user support and phone/mail support functionality will be explored further as the physical database analysis matures in the future.

5.1 HW Design Drivers

The design of the DMGHW CI is primarily based on the volume of service "pull" that will be generated by the user community and the processing load that is placed on the Advertising Service CI and Gateway CI DBMSs as a result. The following is a list of the major contributing sources that drive the design of the DMGHW CI:

- User community access profiles as predicted by User Modeling analysis
- Data Modeling and Data Architecture analysis depicting what classes of information are held, and within what system components
- COTS software selection of DBMS and HTTP servers
- Core system functional and performance requirements

In short, the DMGHW CI must support user demand for: 1) V0 data access, 2) World Wide Web services 3) DBMS services. Design recommendations provided at this time are geared toward Release A; however, scalability, migration and evolvability issues for Release B are addressed.

5.1.1 Key Trade-off Studies and Prototypes

The following Trade Studies and/or Technical Papers were used as follows:

- 1) User Pull Analysis Notebook. Defined rate (number per minute) of user search requests across all DAACs (60-TP-004-001).
- 2) Distributed Database Architecture of Data Management Subsystem - Technical Paper. Defined approach for implementing a distributed database architecture across multiple DAACs (430-TP-004-001).
- 3) DBMS Benchmark Report - Technical Paper. Provides performance evaluation of DBMS COTS packages (430-TP-003-001).

5.1.1.1 Prototype Studies

The following prototypes are pertinent to the Release A Data Management HWCI design:

- Advertising Service CI Prototype: At this time limited performance evaluations have been performed on the Advertising CI prototype; however, one of the goals for the Advertising CI EP6 prototype will be to obtain realistic performance characteristics and use them as inputs to the Hardware Design.
- Gateway CI Prototype: The first Gateway CI prototype will be functional in the CDR time frame; therefore, one of the goals (post CDR) will be to obtain realistic performance characteristics and use them as inputs to the Hardware Design.

5.1.2 Sizing and Performance Analysis

At this time, preliminary performance analysis data is being used to calculate the size of the DMGHW CI DBMS server. The performance data is derived from User Modeling, Vendor, and DBMS Technical Paper inputs. As the Incremental Track software design matures and is subjected to future prototypes, performance results will be revised accordingly. The operations support workstations are being sized according to operations staffing and functionality requirements at each DAAC site.

Processor Capacity Sizing:

Processor sizing for the DMGHW CI is dependent on Incremental Track Development prototyping, vendor inputs for processor ratings in Transactions Per Second (TPS), and User Modeling data for rate of search requests (transactions) at each DAAC site. A performance profile of the transactions to be performed as a function of the Advertising Service CI is to be determined as part of the EP6 prototype (see Advertising Service CI prototype). A performance profile of the transactions to be performed as a function of the Gateway CI is to be determined after Release A CDR (see Gateway CI prototype).

Vendor performance inputs are available in the Data Management section of each of the DAAC-specific volumes, for the Release A sites that are configured with operational DM functionality (LaRC, MSFC, GSFC).

User Modeling analyses that provide sizing input to the DMGHW CI include number of concurrent users and/or concurrent service requests per DAAC per time frame, and frequency of searches per DAAC per time frame. Peak and nominal service request arrival rates are also provided by the User Modeling source data. The preliminary DMGHW CI processor capacity sizing has been achieved by deriving the number of transactions per second (in relation to the CPU) to be performed by each individual service type (search request) and multiplying out by the frequency with which the service types are invoked. User Modeling Service classes/types that will be invoked by the user community have been defined by the User Modeling Group. The following is a list of service types that will be supported by the Advertising Service CI and Gateway CI services (as defined by User Modeling):

- Simple search /1
- Simple search /M
- Match-up search /1

- Match-up search /M
- Coincident search /I
- Coincident search /M

The User Modeling analysis results are available in the Data Management section of the DAAC-specific volumes mentioned previously.

5.1.2.1 HWCI Alternatives

The following candidate processor classes have been evaluated for Release A:

- Uni-processor Server—Uni-processor servers are low cost, single CPU designs. Uni-processor servers allow for moderate concurrent user loads, and offer very little in terms of scalability.
- Low-End Symmetric Multiprocessor (SMP)—Low-End SMPs provide excellent scalability (from either 1-2, or 1-4 CPUs), graceful performance degradation, load balancing and excellent cost/performance benefits. SMPs offer higher internal bus bandwidths and are capable of running DBMS processes in parallel across multiple processors.

Disk storage capacity sizing for the DMGHW CI was determined for each DAAC site based on preliminary DBMS sizing efforts for the Advertising Service CI and Gateway CI operational databases plus vendor inputs for the following COTS software: 1) DBMS software, 2) HTTP & WAIS server software, 3) Operating System software, 4) Communications and Utilities software. Estimates for total disk storage capacity is available in the DAAC unique volumes mentioned previously.

5.1.3 Scalability, Evolvability and Migration to Release B

The DMGHW CI will be designed with sufficient scalability in order to make the transition from Release A to Release B a smooth one. The DMGHW CI will support three additional Application CIs in Release B: 1) Data Dictionary CI, 2) Local Information Manager CI, 3) Distributed Information Manager CI. A significant increase in the volume of user traffic will also impact the DMGHW CI in Release B.

Processing:

In terms of processing scalability, a low-end SMP server clearly out performs a Uni-processor server architecture. The Uni-processor architecture is not very flexible and does not offer much in terms of future processing expansion, where as the low-end SMP is very flexible since multiple processors can be added. Since the SMP can be configured with additional processors it offers significant future growth capability and investment protection, because the cost of adding an additional CPU is much less than purchasing an additional server. The SMP architecture is also much more likely to be supported by technology advancements (technology refreshes) that include processor upgrades. For example, companies such as Hewlett Packard offer well defined upgrade paths for their SMP architecture's that include processor upgrades to future chip designs, but many of HP's Uni-processor systems will not be supported by future processor upgrades.

Since DBMS software servers can be run in parallel over multiple processors, it makes much more sense to implement an SMP architecture when a DBMS is the definitive process in the design, such as in the case of the DMGHW CI. Running a DBMS software server in parallel across multiple

processors is dependent on vendor specific software implementations, and is not an automatic function of a DBMS server design.

Implementing a low-end SMP server architecture in Release A is more than adequate for the sites included as part of Release A; however, given that the amount of hits on the Advertising Service CI is largely unknown at this point, an SMP would provide a resource buffer for immediate expansion in processing. The low-end SMP architecture also provides good scalability in relation to the addition of three additional software CIs in Release B: 1) Data Dictionary CI, 2) Local Information Manager CI, 3) Distributed Information Manager. Furthermore, a low-end SMP architecture satisfies the design requirement of 100% growth in capacity with minimal impact to the physical components of the DMGHW CI server. The only way to satisfy this requirement with a Uni-processor architecture is to purchase an additional server, which is not cost effective. Implementing a low-end SMP architecture in Release A also makes the migration to Release B simplistic since the only configuration change would be to add additional processors (if necessary). The impact on the Operations Staff at each Release A DAAC facility; therefore, would be minimal. More than likely, the low-end SMP architecture will satisfy the processing needs of the DMGHW CI server in Release B and beyond. Given that the amount of time between the Release A and Release B missions/operations is relatively short, it is recommended that the DMGHW CI server hardware design be sized, and implemented with Release B design goals in mind. The DMGHW CI migration plan is available in the DAAC unique volumes mentioned previously.

Storage:

The current DMGHW CI design allows for both standard and RAID disk application without rework of the core design. This can be accomplished through the application of channel adapted disk capacity upgrades (e.g., larger RAID units), or through network adapted disk servers addressable by the DBMS servers. These growth decisions, which are likely to occur at the DAAC due to science mission growth and/or expansion, will be made through predictions on future DBMS transaction rate needs, as well as I/O and storage requirements.

5.2 HWCI Structure

The DMGHW CI consists of four major components that support DBMS processing, DBMS management, data specialists and user support functions. A block diagram for the Release A DMGHW CI configuration is shown in Figure 5.2-1. The block diagram illustrates the physical layout of the DMGHW CI and the interconnection between the different components.

5.2.1 Connectivity

Local Disk Connectivity:

The DMGHW CI servers and workstations will be utilizing host attached disk drives that are connected via SCSI-2 for Release A and possibly Fast/Wide SCSI for Release B.

Network Connectivity:

The DMGHW CI servers and workstations will be directly connected to the local DAAC FDDI network. The DMGHW CI hosts will be connected to the same FDDI ring as the Data Server hosts, as is illustrated in the Data Management Network Connectivity Diagram (Figure 5.2.1-1). Each DMGHW CI host will contain dual-attached station (DAS) cards, which will be dual-homed to separate FDDI concentrators. This provides redundancy so that full connectivity will exist to the

DMGHW CI servers in the event of a concentrator failure. The workstations will contain single-attached station (SAS) cards and each will be connected to a single concentrator, but they will be split across the FDDI concentrators so that they are not all connected to the same concentrator. The FDDI concentrators are, in turn, connected to the local DAAC FDDI switch (Refer to Section 5.2 of Volume 0 for a general description of DAAC networks). A diagram depicting how the DMGHW CI will be connected to the local DAAC FDDI network is shown in the following network connectivity diagram (Figure 5.2.1-1).

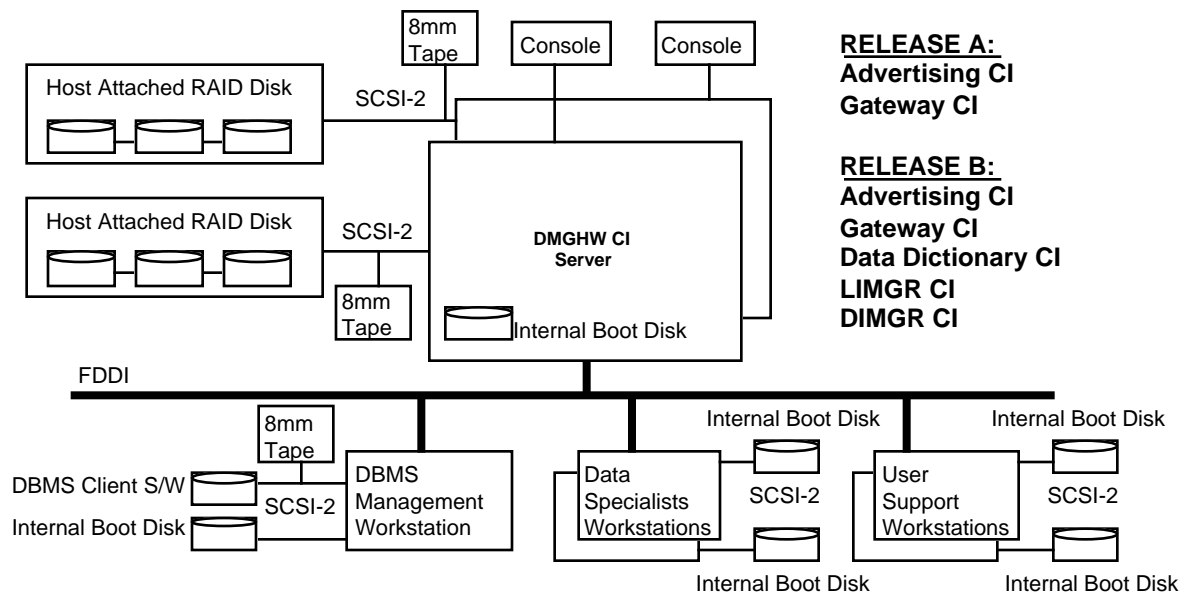


Figure 5.2-1. Data Management HWCI

5.2.2 HWCI Components

The hardware associated with the DMGHW CI consists of servers, low-end uni-processor workstations, RAID disk, and 8mm tape drives used in support of Release A Advertising Service CI and Gateway CI database configurations. The number of physical components, and whether or not certain components will be used, is dependent on each DAAC specific configuration. Table 5.2.2-1 provides an overview of the major physical components of the DMGHW CI.

5.3 Failover and Recovery Strategy

The DMGHW CI server will meet Reliability, Maintainability and Availability requirements as stated in "Availability Models/Predictions for the ECS Project" (515-CD-001-003), "Maintainability Predictions for the ECS Project" (518-CD-001-003), and "Reliability Predictions for the ECS Project" (516-CD-001-003) documentation. The DMGHW CI will be designed to meet the following RMA requirements for Release A and Release B:

- (1) Availability: 0.993
- (2) Mean Down Time: <2 Hrs.

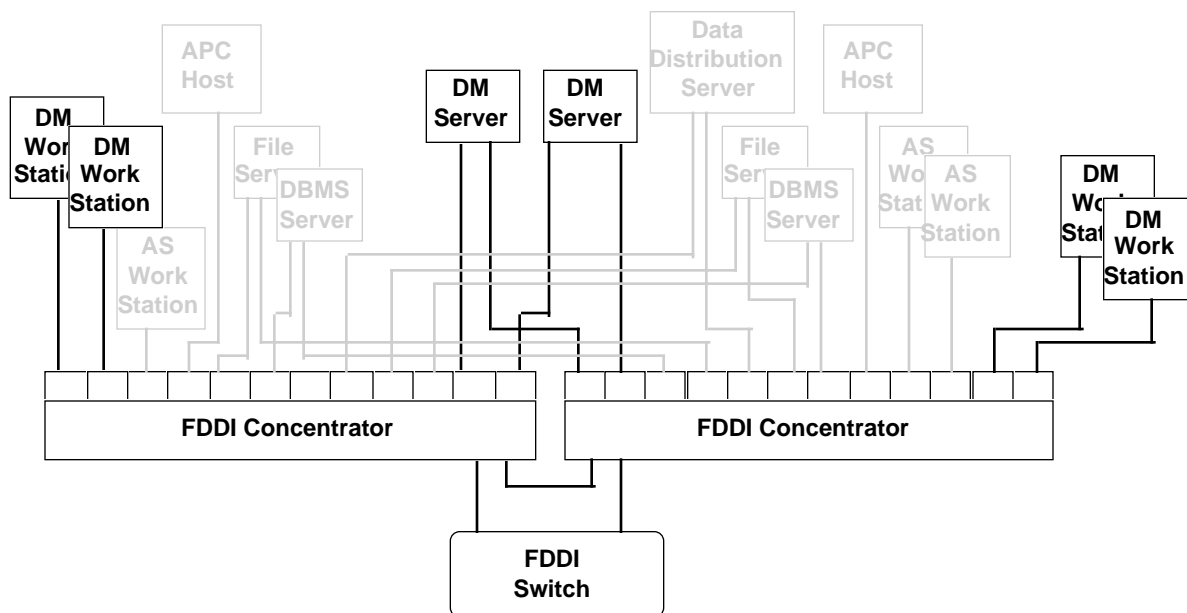


Figure 5.2.1-1. Data Management Network Connectivity

Table 5.2.2-1. Data Management HWCI Components

Component Name	Class/Type	Comments
DBMS/Web Server (Processing)	High-End Uni-processor Workstation Server	Designed to meet small to moderate processing loads. May be used at some sites depending on volume of user (search) traffic. Would provide processing, I/O and disk resources in support of the Advertising Service and Gateway CI's in Release A.
	Low-End SMP Server (1-2, or 1-4 CPUs	Designed to be versatile with regard to scalability, and meet moderate to high processing loads. Used at sites with moderate to heavy volume of user (search) traffic. Would provide processing, I/O and disk resources in support of the Advertising Service and Gateway CI's in Release A.
	Host Attached RAID Disk	RAID disk used for storage in support of Advertising Service CI and Gateway CI DBMS software configurations. Designed for high availability, but not necessarily required for Release A due to the static nature of the Advertising and Gateway CI databases.
	8mm Tape Drive	8mm tape drive used to backup Advertising Service CI and Gateway CI databases and perform other routine maintenance functions. One 5-7 GB unit connected to each DBMS server host.
DBMS Management Workstation	Low-End Uni-processor Workstation	Designed to meet small processing loads. Used to support management activities to be performed on the Advertising Service and Gateway CI databases by the DataBase Administrator (DBA). Will provide processing, I/O and disk resources to the DBA.
	8mm Tape Drive	8mm tape drive used to perform routine DBA maintenance functions (One 5-7 GB).
Data Specialist Workstations	Low-End Uni-processor Workstations	Used in support of Data Specialist functions.
User Support Workstations	Low-End Uni-processor Workstations	Used in support of User Support functions.

5.3.1 Server Hardware Failure Recovery

The DMGHW CI will contain two DBMS servers in Release A. One server will be the primary, or "active" server and the other will be the secondary, or "standby" server. In the event of a hardware failure on the active server, the standby server will be configured so that it can readily assume the processing functions, or role of the active server so as to meet availability requirements. The failure recovery strategy for Release A; therefore, is simply to activate the standby server in the event of a failure to the active server. For Release B, the DMGHW CI can be configured in a redundant, or "active-active" host cluster configuration. The DMGHW CI server cluster can be configured such that in the event of a processor, or local bus (exclusive of network), failure to the primary host, the secondary host will assume the processing "role" of the primary host. The automatic failover / recovery is achieved via communications protocols and system processes that bind the hosts together and allow them to provide excellent levels of availability (and flexibility) in support of mission critical applications. The automatic failover / recovery design will protect mission critical applications from a wide variety of hardware and software failures. For example, the following failures are readily detected and responded to: 1) System processors, 2) System memory, 3) LAN adapters, 4) System processes, 5) Application processes. The design that has been described includes no single point of failure assuming that data disk drives are mirrored and multiple LAN interconnects are used (for example dual homed FDDI). The main advantage to implementing the "active-active" host cluster configuration for Release B is the ability to run processes in parallel across both cluster hosts. Failure and recovery configurations for the DMGHW CI are discussed further in the DAAC unique volumes mentioned previously.

5.3.2 DBMS Failure Recovery

The DMGHW CI will house both Advertising Service CI and Gateway CI databases in Release A. DBMS recovery techniques are currently under analysis. DBMS Replication Server strategies, along with mirrored disk strategies, are being scrutinized and will continue to be explored further as the physical database analysis matures in the future. Both the Advertising Service CI database and the Gateway CI database will be backed up onto tape media on a regular basis to ensure a complete recovery capability.

Disk Mirroring:

Disk mirroring provides a form of redundancy to protect against hardware failure and to provide a degree of fault tolerance. Disk mirroring is the capability to maintain a replicate of all data stored on a disk media device. Disk mirroring can provide non-stop recovery in the event of a disk failure. With respect to disk failure recovery, the DMGHW CI servers will have multiple physical disks to service safe storage of the production DBMS. The disks will be grouped into two logical sets and the second set will mirror the first set. There are two basic strategies that can be employed:

- (1) *Hardware supported*—physical cross strapping of disk between two servers, with one acting as the primary and the second acting as the backup, or mirror set.
- (2) *Primarily software based*—disk mirroring between a primary and a secondary server through the use of protocols and LAN networks.

Processes such as Sybase replication server or Sybase disk mirroring capability can also be executed to provide disk mirroring capability. A Sybase SQL server database device can be duplicated, i.e. all writes to the device are copied to a separate physical device so that if one of the devices fail,

the other contains an up-to date copy of all transactions. Sybase replication server and disk mirroring capability is currently under evaluation to determine if a method of deployment is necessary given RAID disk alternatives.

When deciding to mirror, one must weigh factors such as the cost of system downtime, possible degradation of performance, and the cost of storage media. Considering these issues, one has to decide what devices to mirror; selected devices such as transaction logs, or all devices. Mirroring selected devices minimizes disk resources and performance degradation but requires manual intervention to restore the un-mirrored devices from backup in the event of a disk hardware failure. Mirroring all disk devices such as the master device, user databases, and transaction logs provides a non-stop recovery from hardware failure, but provides a slight degradation in performance. In the event of disk media failure, the mirror device can take over, typically without any downtime. When the damaged device is repaired or replaced, it can then be re-synchronized with the undamaged, operational device.

5.3.3 Network Recovery

There are three types of network failures that may affect the DMGHW CI: 1) A FDDI cable failure due to physical damage would require a new cable to be installed, 2) If an individual port on the FDDI concentrator fails, then the attached host would have to be routed to another port, 3) If the entire FDDI concentrator fails, then it would have to be replaced (which can be done rapidly since the units require very little configuration).

Note that the above failures result in service interruption only to the workstations. Since the DBMS server hosts are attached to separate concentrators via dual-attached station (DAS) cards, they will communicate as normal in the event of a single cable, or single concentrator fault; therefore, critical applications would not be affected by the failure.

Abbreviations and Acronyms

ACMHW	Access Control and Management HWCI
ADC	Affiliated Data Center
ADS	Archive data sets
ADSHW	Advertising Service HWCI
ADSRV	Advertising Service CSCI
AITHW	Algorithm Integration & Test HWCI
AITTL	Algorithm Integration and Test Tools (CSCI)
AM	Ante meridian
ANSI	American National Standards Institute
APC	Access/Process Coordinators
API	Application Programming Interface
APID	Application Process Identifier
AQAHW	Algorithm QA HWCI
ASAP	As soon as possible
ASCII	American Standard Code for Information Interchange
ASF	Alaska SAR Facility (DAAC)
ATM	Asynchronous Transfer Mode
CD ROM	Compact disk read only memory
CDRL	Contract Data Requirements List
CERES	Clouds and Earth's Radiant Energy System
CI	Configuration Item
CIESIN	Consortium for International Earth Science Information Network
CLS	Client Subsystem
COTS	Commercial off-the-shelf
CPU	Central processing unit
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
CCSDS	Consultative Committee for Space Data Systems
CM	Configuration Management
CSDT	Computer Science Data Types
CSMS	Communications and Systems Management Segment
CSS	Communication Subsystem (CSMS)

DAA	DAN Acknowledge
DAAC	Distributed Active Archive Center
DADS	Data Archive and Distribution System
DAN	Data Availability Notice
DAO	Data Assimilation Office
DAR	Data Acquisition Request
DAS	Data Availability Schedule
DBA	Database administrator
DBMS	Database Management System
DDA	Data Delivery Acknowledgement
DDICT	Data Dictionary CSCI
DDIST	Data Distribution CSCI
DDN	Data Delivery Notice
DDSRV	Document Data Server CSCI
DESKT	Desktop CI
DEV	Developed code
DID	Data Item Description
DIM	Distributed Information Manager
DIMGR	Distributed Information Management CSCI
DIPHW	Distribution & Ingest Peripheral Management HWCI
DMGHW	Data Management HWCI
DMS	Data Management System
DMS	Data Management Subsystem
DP	Data Processing
DPR	December Progress Review
DPREP	Science Data Pre-Processing CSCI
DPS	Data Processing Subsystem
DR	Data Repository
DRPHW	Data Repository HWCI
DS	Data Server
DSM	Distribution Storage Management
DSS	Data Server Subsystem
DT	Data Type
ECS	EOSDIS Core System
EDC	EROS Data Center (DAAC)
EDOS	EOS Data and Operations System

EOC	Earth Observation Center (Japan)
EOS	Earth Observing System
EOSDIS	Earth Observing System Data and Information System
EP	Evaluation Package
EP	Early Prototype
ESDIS	Earth Science Data and Information System
ESDT	Earth Science Data Types
F&PRS	Functional and Performance Requirements Specification
FC	Fiber Channel
FDDI	Fiber distributed data interface
FDF	Flight Dynamics Facility
FOS	Flight Operations Segment
FSMS	File and Storage Management System
Ftp	File transfer protocol
GB	Gigabyte
GDAO	GSFC Data Assimilation Office
GFLOPS	Giga (billions) Floating Point Operations per Second
GOES	Geostationary Operational Environmental Satellite
GRIB	Gridded Binary
GSFC	Goddard Space Flight Center
GTWAY	Version 0 Interoperability Gateway CSCI
GUI	Graphic user interface
HDF	Hierarchical Data Format
HiPPI	High Performance Parallel Interface
HMI	Human machine interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transport Protocol
HWCI	Hardware Configuration Item
I&T	Integration and Test
I/O	Input/Output
ICD	Interface Control Document
ICLHW	Ingest Client HWCI
IDL	Interface Definition Language
IEEE	Institute of Electrical and Electronics Engineers
IERS	International Earth Rotation Service
IMS	Information Management Subsystem

IP	International Partner
IR-1	Interim Release 1
IRD	Interface Requirements Document
IS	Ingest Subsystem
ISS	Internetworking Subsystem (CSMS)
JPL	Jet Propulsion Laboratories
LaRC	Langley Research Center
LIM	Local Information Manager
LIMGR	Local Information Management CSCI
LIS	Lightning Imaging Sensor
L0	Level 0
MB	Megabyte
Mbps	Megabits per second
MBps	Megabytes per second
MD	Maryland
MFLOP	Millions of Floating Point Operations per Second
MOC	Mission Operations Center
MODIS	Moderate-Resolution Imaging Spectrometer
MPP	Massively Parallel Processor
MRF	Medium Range Forecast
MSFC	Marshall Space Flight Center
MSS	Management Subsystem (CSMS)
MTBF	Mean time between failures
MTTR	Mean time to restore
NESDIS	National Environmental Satellite Data and Information Service
NMC	National Meteorological Center
NOAA	National Oceanic and Atmospheric Administration
NSIDC	National Snow and Ice Data Center (DAAC)
O/A	Orbit/Attitude
ODC	Other Data Center
ODL	Object Description Language
ORNL	Oak Ridge National Laboratory (DAAC)
OSM	Open Storage Manager
OTS	Off-the-shelf
PAM	Permanent Archive Manager
PCI	Periphewral Component Interface

PDPS	Planning and Data Processing System
PDR	Preliminary Design Review
PDS	Production Data Set
PDS	Production Data Specialist
PGE	Product Generation Executive
PGS	Product Generation System
PLNHW	Planning HWCI
POSIX	Portable Operating System for UNIX
PRONG	Processing CSCI
Q	Quarter
Q/A	Quality Assurance
QA	Quality Assurance
QAC	Quality and Accounting Capsule
RAID	Redundant Array of Inexpensive Disks
RAM	Random Access Memory
REL	Release
RID	Review Item Discrepancy
RMA	Reliability, Maintainability, Availability
RTF	Rich Text Format
S/C	Spacecraft
SAA	Satellite Active Archives (NOAA)
SCF	Science Computing Facility
SCSI II	Small Computer System Interface
SDF	Software Development File
SDP	Science Data Processing
SDPF	Sensor Data Processing Facility (GSFC)
SDPS	Science Data Processing Segment
SDPS/W	Science Data Processing Software
SDPTK	SDP Toolkit CSCI
SDSRV	Science Data Server CSCI
SFDU	Standard Format Data Unit
SMC	System Monitoring and Coordination Center
SMP	Symmetric Multi-Processor
SPRHW	Science Processing HWCI
STMGT	Storage Management CSCI
TBD	To be determined

TBR	To be resolved
TDRSS	Tracking and Data Relay Satellite System
TONS	TDRSS Onboard Navigation System
TRMM	Tropical Rainfall Measuring Mission
TSDIS	TRMM Science Data and Information System
UR	Universal Reference
USNO	United States Naval Observatory
V0	Version 0
VC	Virtual Channel
VCDU-ID	Virtual Channel ID
WAIS	Wide Area Information Servers
WAN	Wide Area Network
WKBCH	Workbench CI
WKSHC	Working Storage HWCI
W/S	Workstation
WORM	Write Once Read Many
WS	Working Storage
WWW	World Wide Web